NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Application of a logical inference mechanism to queries of legal documents

Jean G. LAIHUNG

A Thesis

in

The Department

of

Computer Science

Presented in Partial fulfillment of the Requirements
for the Degree of Master of Computer Science at
Concordia University
Montréal, Québec, Canada

February 1992

ISBN 0-315-73693-3

Canada

Abstract

Application of a logical inference mechanism to queries of legal documents

Jean G. LAIHUNG

The user of legal documents, usually, does not know the relationships between the legal terms. If he has some problems which need to satisfy the legal documents, he has to consult a human expert. We want to set up a computer-aided problem solving mechanism related to legal documents. For this purpose, the logical inference mechanism of Prolog seems one of the ideal methods. With our system, we wish to solve some specific feature which is not very well treated in existing expert systems.

Legal code clauses do not always fit Horn clauses of Prolog directly. We show one method to represent and handle legal code clauses, using the National Building Code of Canada as an example. The menu-driven system implements the forwar chaining mechanism and the logical inference mechanism in Quintus Prolog. For the former one, a prototype system shell has been developed, and enables the user to find the code regulations by choosing the physical constraints related to the building, according to the menu. For the latter, a logical query system has been implemented, and enables the user to find all possible answers to a given

query under some physical constraints arbitrarily defined by the user. The latter is used when the user has only a partial knowledge of the physical constraints related to the building, and the logical inference system will find all possible solutions which match the partial knowledge of the user. This logical inference mechanism is applicable, in general, for all legal codes. When the user has a partial knowledge of the parameters related to the query, he will get possible answers and possible values of parameters which were not known before.

This approach is compared with previous works in which decision tables and production systems have been used for processing legal codes. We propose to combine our problem-solving mechanism and existing expert systems as a future work.

Acknowledgements

I would like to acknowledge the invaluable assistance and advice of my thesis supervisor, Dr M. Okada, in the preparation of this thesis.

I would like to express my thanks to the thesis examining committee members. Their advice have greatly improved the final form of the thesis.

TABLE OF CONTENTS

LIST OF FIGURES AND TABLES

CHAPTER 1

INTRODUCTION

1.1 Objective

The user of legal documents, usually, does not know the relationships between the legal terms. If he has some problems which need to satisfy the legal documents, he has to consult a human expert. We want to set up a computer-aided problem solving mechanism related to legal documents. We wish to solve some specific feature which is not very well treated in existing expert systems.

The objective of this thesis is to show the application of logical inference mechanism to queries of legal documents using the National Building Code of Canada [1] as an example. The National Building Code of Canada (1990) is the official document implementing regulations for the design and construction of buildings. The regulations are dependent on the use and classification of the building. Quintus Prolog [2] was chosen as the programming language as it contains an inherent inference engine, which is not available in traditional procedural or functional languages. The prototype system uses a menu driven mechanism in a blackboard problem-solving environment. The proposed method for partitioning the knowledge bases makes them more understandable, maintainable, efficient, and suitable for

parallel processing. Changes to a prolog program are extremely simple. Since knowledge exists as declarative statements, changing any existing knowledge or adding new knowledge can easily be performed.

The system is described in chapter 3, and should be considered as an advice system which can be used as an aid to the design process. The system is an advance in the automation of the engineering design process. The proposed system supports not only the forward-chaining but the logical inference mechanisms.

Logical inference mechanism described in chapter 4, is based on Horn clause logic, and is shown to be the most appropriate and elegant method to extract the relationships between the variables and constraints in the design codes. Examples are given to display the expressive power of logic to find the relationship between the number of storeys and building area for fire regulation clauses in the code. By using the information about the user constraints and preferences, the system proposes the code clauses that conform to all constraints.

1.2 Scope

The program implements the major regulations applying to Part 3 of the National Building Code. The size and occupancy requirements for fire safety, the safety requirements within

floor areas, requirements for exits, and health requirements as applied to plumbing facilities are implemented.

The system runs on the Apollo workstation under the Unix operating system.

Rosenman and Gero [4] developed an expert system shell for compliance checking of the Australian Model Uniform Building Code (AMUBC). Their shell was first written in Prolog 1 (from Expert Systems Limited) running on an 8088/8086 based microcomputer in an MS-DOS environment, and then later implemented in Quintus Prolog.

With the help of the proposed system, the user can choose one of the menu items available, i.e, building identification, fire regulations, fire alarm requirements, term definitions, safety requirements within floor areas, requirements for exits, plumbing facilities. Each menu item corresponds to a subset of clauses in the code. The user can get the code requirements by answering a series of questions. The system saves the answers in a working database which remembers all the answers and temporary information during a given consultation.

The important feature of the system is the ability of the user to extract all relationships between data items by choosing the menu item 'query'. A user-interface written in Prolog, gives in an English-like language, all the solutions to a query based on the user's constraints. For example, when you are designing an arena facing two streets, you want

to know if or not you need to install a sprinkler or you want to know the precise relation between the sprinkler regulation and the possible size of the building. Then under the given constraints, say a corner of two streets and an arena, the logical engine tries to find all the possible conditions on the sprinkler and the building area. Or, when you do not want to install sprinklers for some reason (say a budget reason, or some local difficulty of water system) and you want to build an arena, you want to know in what kind of location and how many storeys you can build. The same logical engine can solve this query.

The main menu of the system is as follows :

```
1  Building code regulations
2  Query with constraints
3  Exit to unix
4  Exit to prolog
Your choice (1 to 4) : 2
```

If the user knows all the physical constraints of the building such as the number of storeys, the building area in m2, he can choose item 1. The system asks yes-no questions regarding his constraints and the system will give him the applicable regulations of the building code. This is the traditional process of forward chaining mechanism.

If the user does not know all the constraints, he chooses item 2. For example, the user wants to build an arena and he wants to find the fire regulations in the building code. For the fire regulations, given physical constraints, the code gives the required fire resistance rating of the structural components of the building. Assume the user chooses item 2. The system responds with the query menu.

```
Menu 2

1 Fire Regulation
2 Fire Alarm
3 Safety Requirements within floor areas
4 Requirements for exits
5 Plumbing facilities
6 Previous menu

Your choice (1 to 6) : 1
```

When the user chooses the item ' Fire Regulation ', the system presents him with the following menu :

```
Query : Fire Regulation

Rules are defined as rule(A,S,X,Y,Z)

Use the following variables

X = number of storeys

Y = building area in m2
Z = number of streets the building faces
A = rule number
S = status whether the building is sprinklered or not
S = 1  (unsprinklered)

S = 2  (sprinklered)
```

```
: rule(A,1,1,Y,2).
```

In our example, the given constraints are the following :

the arena is unsprinklered (variable S = 1),
it is one-storey (variable X = 1),
it faces two streets (variable Z = 2).
The user must apply the format rule(A,S,X,Y,Z) for his query. In this example, the user inputs the following :

rule(A,1,1,Y,2).

The system finds from the user, the building identification by asking yes-no questions. The interactive session is as follows :

Is the building used for assembly occupancies?
e.g, theatre, art gallery, church, club, court room, dance hall, day-care centre, gymnasium, lecture hall, library, licensed beverage establishment, lodge room museum, passenger station, depot, recreational pier, restaurant, school, nonresidential undertaking premise
> y

Is the building used for the production and viewing of the performing arts?
> n

Is the building used as an arena?
e.g,ice rink, indoor swimming pool
> y

Is the building used for the congregation or gathering of persons participating in or viewing open air activities? e.g, amusement park, bleach, grandstand, stadium.

> **n**

The system displays the first solution. The solution found in this example, is the maximum building area based on the material properties (fire rating) of the structural components of the building such as floor assemblies. Now the first solution is the following :

Solution found: rule(20,1,1,1250,2)

Number of storeys = 1

Building area = 1250 m2 max

Building is unsprinklered

Building faces 2 streets

Rule 20 (provision 3.2.2.24)

Fire regulation

Building shall be of combustible or noncombustible construction used singly or in combination.

The original constraints given by the user are not underlined. The underlined are other related constraints which were found by the logical inference system. For example, the other related constraint is the building area

equal to 1250 m2 max. The system asks the user if he wants another solution :

Look for another? (Y/N) : Y

Then the second solution comes up if any. In our example, the system finds another solution with another constraint of the building area.

Solution found: rule(21,1,1,3000,2)
Number of storeys = 1
Building area = 3000 m2 max
Building is unsprinklered
Building faces 2 streets

Rule 21 (provision 3.2.2.25)
Fire Regulation
Building shall be of combustible or noncombustible construction used singly or in combination, and the fire-resistance rating (if of combustible construction) for

(a) mezzanines	*> 45 min*
(b) roof assemblies	*> 45 min*
(c) loadbearing walls,columns,arches	*> 45 min*

*Look for another? (Y/N) :***Y**

The user can repeat the same process until the system answers " No more solutions ".

Solution found: rule(22,1,1,5000,2)

Number of storeys = 1

Building area = 5000 m2 max

Building is unsprinklered

Building faces 2 streets

Rule 22 (provision 3.2.2.26)

Fire Regulation

Building shall be of noncombustible construction, and

the fire-resistance rating for

(a) floor assemblies > 1 h

(b) mezzanines > 1 h

(c) roof assemblies > 45 min

(d) loadbearing walls, columns, arches greater than that required for the supported assembly.

Look for another? (Y/N) **:y**

Solution found: rule(23,1,1,unlimited,2)

Number of storeys = 1

Building area = unlimited

Building is unsprinklered

Building faces 2 streets

Rule 23 (provision 3.2.2.27)

Fire Regulation

Building shall be of noncombustible construction, and

the fire-resistance rating for

(a) floor assemblies	*> 2 h*
(b) mezzanines	*> 1 h*
(c) roof assemblies	*> 1 h*
(d) loadbearing walls, columns, arches	*greater than that required for the supported assembly.*

Look for another? (Y/N) : y

No (more) solutions

Now, the user gets the statement " No more solutions ". This information is sometimes very useful because the system provides information about conditions related to the building that are physically possible but impossible according to the code. We shall describe later in section 4.3.

CHAPTER 2

LEGAL CODES

2.1 Codes

Public safety is involved in the design and fabrication of buildings, and, to minimize the danger of catastrophic failure or even premature failure, documents are established to regulate the design and construction of buildings. These documents are called specifications, codes, standards, and rules. Sometimes the terms are used interchangeably. Webster's Third International Dictionary (1969) defines the term Code as follows :

Code : " A set of rules of procedure and standards of materials designed to secure uniformity and to protect the public interest in such matters as building construction and public health, established usually by a public agency".

Codes and specifications are generally written by professional organizations, or government bureaus, and each code deals with applications pertaining specifically to the interest of the authoring body. Some specifications rigidly call out the design and fabrication procedures to be followed and are legally binding. Meeting the requirements of a code does not protect anyone against liability concerning the performance of the structure. Nor, in

general, does any code-writing body approve, endorse, guarantee, or in any way attest to the correctness of the procedures, designs, or materials selected for code application. Now, we consider the National Building Code as an example.

2.2 National Building Code

The National Building Code of Canada (1990) sets standards and regulations on buildings for the protection of public rights and for the dissemination of empirical knowledge gathered over a long period of time. Since these regulations have been written by various authorities in different disciplines, they are likely to be ill structured. Consequently, finding and interpreting the regulations is time-consuming and difficult.

The knowledge in the National Building Code is deterministic. No probability exists in the truth values of the requirements for a given set of truth values of the conditions. In this thesis, we shall concentrate on the fire regulation part of the National Building Code (section 3.2.2 to 3.2.3), because we believe that this is a good example which users want to consult before they design a building. The fire regulation provisions of the building code specify the needs of sprinklers, and the required fire resistance rating of the structural components of the building, for some physical constraints such as the number

of storeys,and the use of the building.

Requirements are determined by the designer before the building is designed. The National Building Code classifies buildings according to the following list :

(a) Group A , assembly occupancy

(i) Division 1,

(ii) Division 2,

(iii) Division 3, or

(iv) Division 4,

(b) Group B, institutional occupancy,

(i) Division 1, or

(ii) Division 2,

(c) Group C, residential occupancy,

(d) Group D, business and personal services occupancy

(e) Group E, mercantile occupancy, or

(f) Group F, industrial occupancy,

(i) Division 1,

(ii) Division 2, or

(iii) Division 3.

The user may wish to find the requirement for the type of construction of the various elements of a particular building. The National Building Code determines the type of construction based on the classification and height of the building, and whether the building is sprinklered or not.The user must determine for himself the particular regulations applicable to his requirements.

Building regulations are based on empirical and heuristic knowledge. A knowledge-based system methodology can be applied to solve the non-sequential procedures that cannot be efficiently handled by procedural programming languages.

2.3 Typical Form of the legal code clause and its relation to Prolog's Horn clause

Example 1 on page 16 shows an example of the fire regulation provision in the building code. The provision has the following format. *Sub-section 1* gives the conditions of the provision, which are usually represented by tables such as *Table 3.2.2.B* in *section 3.2.2.26*. The table displays the relationships among the physical constraints of the building, such as the number of storeys. The corresponding program source code as the body of Prolog's Horn clause is given on pages xvii and lxxvii of the Appendix. *Sub-section 2* shows the consequence of the provision. The consequence

consists of several litterals related to the fire resistance rating of the structural components such as the floor assemblies. Sometimes, there is another exception clause within *sub-section 2*. We treat the exception part by writing another rule. The corresponding program source code for the exception in *sub-section 3.2.2.25* in example 2 on page 17 is given on page xvii of the Appendix. The legal provision can be transformed into Prolog's Horn clause with careful treatment of the table representation and the exception parts of the provisions.

3.2.2.26. Assembly Buildings, Division 3, 1 and 2 Storeys

(1) A *building* classified as Group A, Division 3 shall conform to Sentence (2) provided the *building*

(a) is not more than 2 *storeys* in *building height*,

(b) if unsprinklered, has a *building area* not more than the value in Table 3.2.2.B., and

(c) if *sprinklered*, is not more than twice the area limits of Clause (b)

Table 3.2.2.B.
Forming Part of Sentence 3.2.2.26.(1)

No. of *Storeys*	Unsprinklered Maximum Area, m²		
	Facing 1 *Street*	Facing 2 *Streets*	Facing 3 *Streets*
1	4 000	5 000	6 000
2	2 000	2 500	3 000
Column 1	2	3	4

(2) Except as provided in Clauses (c) and (d), the *building* shall be of *noncombustible construction*, and

(a) floor assemblies shall be *fire separations* with a *fire-resistance rating* of not less than 1 h,

(b) *mezzanines* shall have a *fire-resistance rating* of not less than 1 h,

(c) roof assemblies shall have a *fire-resistance rating* of not less than 45 min or be of *heavy timber construction*, and

(d) all *loadbearing* walls, columns and arches shall have a *fire-resistance rating* not less than that required for the supported assembly, except that arches are permitted to be of *heavy timber construction*.

Reference : NATIONAL BUILDING CODE OF CANADA 1990
National Research Council of Canada,
Ottawa, Ontario

Example of legal clause (1)

3.2.2.25. Assembly Buildings, Division 3, 1 Storey

(1) A *building* classified as Group A, Division 3 shall conform to Sentence (2) provided the *building*

(a) is not more than 1 *storey* in *building height*,

(b) if unsprinklered, has a *building area* not more than

(i) 2 400 m² if facing 1 *street*,

(ii) 3 000 m² if facing 2 *streets*, or

(iii) 3 600 m² if facing 3 *streets*, and

(c) if *sprinklered*, is not more than twice the area limits of Clause (b).

(2) The *building* shall be of *combustible* or *noncombustible construction* used either singly or in combination, and

(a) *mezzanines* shall have, if of *combustible construction*, a *fire-resistance rating* of not less than 45 min,

(b) roof assemblies shall have, if of *combustible construction*, a *fire-resistance rating* of not less than 45 min, except that the *fire-resistance rating* is permitted to be waived provided that the roof assembly is constructed as a *fire-retardant treated wood* roof system conforming to Article 3.1.14.1., and

(i) if unsprinklered, the *building area* is not more than

1 200 m² if facing 1 *street*,

1 500 m² if facing 2 *streets*, or

1 800 m² if facing 3 *streets*, and

(ii) if *sprinklered*, the *building area* is not more than twice the area limits of Subclause (i) (see Article 3.2.2.12 for supervised sprinkler systems), and

(c) all *loadbearing* walls, columns and arches supporting an assembly required to have a *fire-resistance rating* shall have a *fire-resistance rating* of not less than 45 min or shall be of *noncombustible construction*.

(See also Article 3.2.2.24.)

REFERENCE : NATIONAL BUILDING CODE OF CANADA 1990
National Research Council of Canada,
Ottawa, Ontario

Example of legal clause (2)

CHAPTER 3

FORWARD CHAINING MECHANISM AND EXPERT SYSTEMS

3.1 Forward chaining mechanism

The traditional forward chaining mechanism for expert systems can be used when the user knows all the parameters related to a particular problem. In our case, the forward chaining mechanism is applicable when the user knows all the physical constraints of the building such as the number of storeys, and the building area in m2. The system asks yes-no questions regarding his constraints and the system will give him the applicable regulations of the building code.

Experts often serve as consultants. A client comes to the expert with some problem. Through consultation, the expert provides the client with one or both of two basic services : diagnosis and prescription. The client makes the final decision whether to accept the diagnosis and whether to act on the prescription.

How can we get a computer to do the things the consultant does? It will have to ask questions, report conclusions, and explain its conclusions. To reach its conclusions, it will have to contain some internal representation of expert knowledge, and it will have to be able to use this expert knowledge together with the knowledge supplied by the user to arrive at a diagnosis and possibly a solution to the

user's problem. To do this, every expert consulting system must have at least three basic subsystems.

First, there is the part that holds the expert knowledge. We call this the knowledge base of the system.

Second, there is the part of the system that uses the knnowledge base and information provided by the client to arrive at its conclusions. We call this the inference engine. The knowledge base is just a set of electrical signals to represent knowledge, they must be understood and used by the system in an appropriate way. It is the inference engine that determines how the signals stored in the computer are understood and used.

Third, there is the part of the system that we will call the user interface. This part of the system asks the user questions and converts the answers into a form the inference engine can understand. When the inference engine reaches a conclusion, the user interface must communicate it to the user. If the user questions a conclusion, the user interface must be able to explain to the user how the inference engine reached its conclusion.

3.2 Expert System Shells for forward chaining mechanism

A number of products are being offered commercially as expert system shells. Unlike a complete expert system, a shell lacks one of the three key ingredients of an expert system : it has no knowledge base. Expensive expert system shells provide inference engines with a wide range of

capabilities and provide user interfaces with great flexibility. However, a few of the features these systems provide may be needed, but those systems possess many more features that will never be used.

Fazio et al [5] have reviewed the commercial software development tools for design applications. They have compiled Table 1 [5] which shows a qualitative comparison of seven commercial tools.

Table 1 Comparison of knowledge-based system development tools for design applications

Name of Development tool	Knowledge representation methodology	Inference mechanism	Computational capabilities	Data base Interface	External program Interface	Developer interface & debugging facilities	End-user interface development capabilities	Remarks
OPS5	Production rules	Forward chaining	Simple arithmetic	Not available	Not available	Trace available for debugging	Not available	Many of the features are dependent on the particular implementation of OPS5
GURU	Production rules	Forward chaining backward chaining & a combination	Full set of programming features	Integrated	Not available	Command and menu interface, and trace available	Not available	
PC PLUS	Production rules	Backward chaining with option for antecedent rules	PC SCHEME functions	DBASE	DOS-CALL	Menu interface, trace, review and playback	Available	
GOLDWORKS	Frames and production rules	Forward chaining backward chaining & a combination	GCLISP functions	DBASE and Lotus 123	DOS-CALL and C/PASCAL	Menu and LISP level developer	Available	5 MB min RAM and 10 MB min disk space
M 1	Production rules	Backward chaining	Simple arithmetic	Not available	C/Assembler	Trace available for debugging	Not available	
INSIGHT 2+	Production rules	Backward chaining	Simple arithmetic	Not available	PASCAL	Menu interface and trace available	Not available	
NEXPERT OBJECT	Frames and production rules	Integrated forward and backward chaining	Simple arithmetic	Customized data base interface	C	Menu/window interface	Not available	MS WINDOWS program required

Note: Uncertain information handling, explanation and reasoning facilities have not been considered in this investigation.

Reference P.Fazio,C.Bedard,K.Gowri:Knowledge-Based System Development Tools for Processing Design Specifications

The earliest knowledge-based system development tool was OPS5, known as Office Production System. HI-RISE [6], a knowledged-based expert system for the preliminary structural design of high rise buildings, was developed in OPS5 using an object-attribute-value tuple for knowledge representation. The main inadequacy as a development tool is the fact that it does not provide any facility for rule editing, syntax verification or debugging.

GURU [7] is a development tool for the business application software market. It provides a ruled-based system, natural language interpreter, database management and spread sheet analysis. However, interfacing with other programming languages , is not possible.

PC PLUS [8] is a rule-based system development tool with an interactive and menu-driven interface to the end-user. Rules are written in ARL (Abbreviated Rule Language) or LISP form.

GOLDWORKS [9] uses frames and rules in a LISP environment, and provides multiple levels of user-interface capabilities.

Fazio et al. concluded that each development tool forces the user to tailor the problem to fit within the capabilities of the tool.

The alternative to using a shell for an expert system development is to build the inference engine and user interface that is required, in some programming language. Prolog is a good choice for this because it comes with a built-in inference engine. Of course, an inference engine

can be built in Lisp, Pascal, Fortran, or any other programming language. But Prolog has the advantage that a sophisticated inference engine is immediately available and ready to use. Furthermore, Prolog provides a rudimentary user interface. We can enter a query, and Prolog will tell us whether it can satisfy the query from its knowledge base. Prolog also offers a basic explanatory facility. To see how Prolog reaches a conclusion, all we need to do is invoke the trace function.

3.3 Programming style for forward chaining mechanism

As far as possible, programs in any language should be modular. This means that the program is broken up into sections that interact in clearly specified ways. In order to find out what a section of the program does, we need only look at that section and, perhaps, some other sections to which it refers explicitly.

Prolog facilitates modular programming because it has no global variables. A variable has a value only as long as a particular clause is being executed; clauses communicate with each other through arguments. If there are no asserts or retracts, the behavior of any predicate is predictable from its definition and the definitions of the predicates it calls. This makes it possible to break Prolog programs up into sections for execution on separate separate central processing units- a major motive behind the choice of Prolog

for the Fifth Generation Project.

The crucial difference between a Prolog knowledge and a conventional database is that, in Prolog, inferred or deduced knowledge has the same status as information stored explicitly in the knowledge base. That is, Prolog will tell the user whether a query succeeds, and if so, with what variable instantiations. It does not normally tell us whether the answer was looked up directly or computed by inference.

Prolog interprets clauses as procedure definitions. As a result, the language has both a declarative semantics and a procedural semantics. Any Prolog knowledge base can be understood declaratively as representing knowledge, or procedurally as prescribing certain computational actions.

Quintus Prolog implementations allow us to divide a program into "modules'. Predicates defined in one module cannot be called from other modules unless they are declared as 'public'.

Prolog also facilitates top-down design. A program can be designed by writing the main procedure first, then filling in the other procedures that it will call. The interchangeability of facts and rules makes it easy to write stubs, or substitutes, for procedures that are to be written later.

Prolog handles all memory management dynamically. That is, it does not set aside space for constants and variables at

the beginning of the program; instead, it allocates space as needed while the program runs. All memory management is done through pointers, which are memory locations that store the addresses of other memory locations.

3.4 The Blackboard Concept

The blackboard concept was the basis for the HEARSAY II speech recognition system developed by Reddy et al. [10] at Carnegie-Mellon University. Nii and Aiello [11] developed the first system for building blackboard systems (AGE) at Stanford University.

Nii [12,13] has given a detailed description of the blackboard methodology and its use in several applications. A blackboard system can be thought of as a framework in which knowledge can be arranged so that it can be distributed to a set of specialists called knowledge sources. Part of the knowledge is encoded on the blackboard, which is the shared portion of the knowledge base through which the knowledge sources communicate.

An application using a blackboard structure can be viewed schematically as shown in Figure 1.

The blackboard model is a conceptual entity for problem solving, and consists of three major components :

1. **knowledge sources.**

The knowledge needed to solve the problem is decomposed into knowledge sources, which are kept independently of each

other.

2. **Blackboard**

Knowledge sources produce changes to the blackboard that lead incrementally to a solution to the problem.

3. **Control**

problem solving strategy

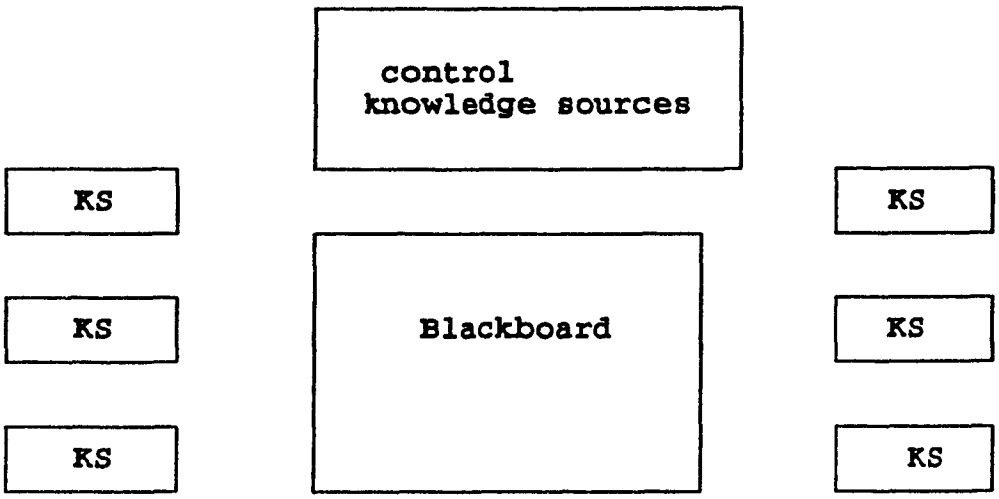


Figure 1 Schematic of a Blackboard System, where KS denotes a knowledge source

The blackboard model makes use of the divide and conquer principle, and decomposition encourages simplification of the problem and enhances efficiency.For instance, a 500-rule system that can logically be divided into 10 sets of 50 rules, is more efficient than the original 500-rule system. This decomposition is particularly useful for applications to be run on personal computers.

Figure 2 shows the **BUILDING** schematic diagram, where the subclauses of the building code are treated as knowledge sources.

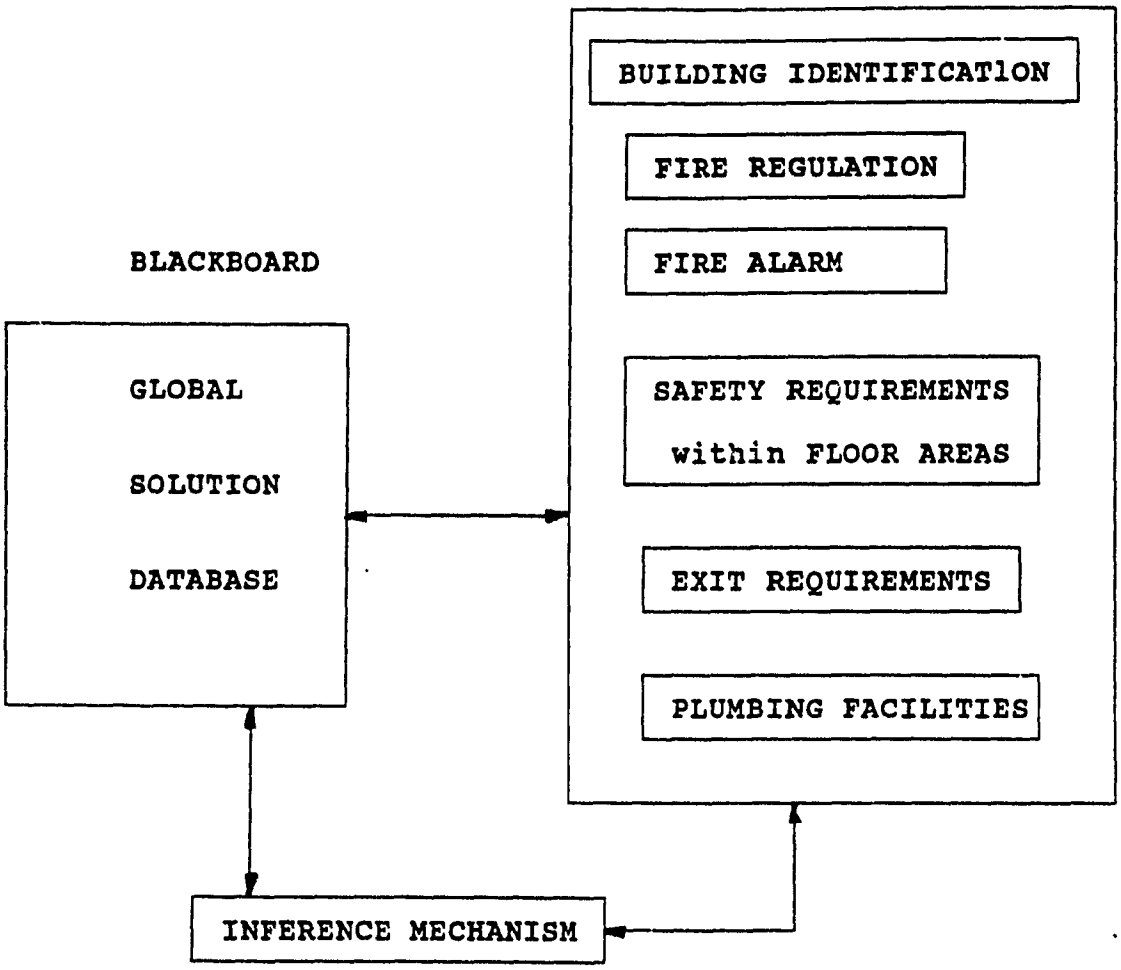


Figure 2 **BUILDING** Schematic Diagram

3.5 XSHELL

Now, we shall explain the implementation of Menu 1, which deals with the forward chaining inference mechanism of the proposed system. We can use either procedural or declarative approaches to programming in Prolog. We will think of XSHELL as a set of procedures for interaction between the user and the Prolog inference engine. Our basic approach in developing XSHELL, then, is procedural. On the other hand, our approach in building knowledge bases to use with XSHELL should be declarative. The idea behind XSHELL is that it provides the auxiliary procedures we need to drive a consultation based on a declaratively developed knowledge base.

XSHELL should :

1. Inform the user what the expert system does and how it is used.
2. Make an identification,remember it,report it,and explain it if required to do so.
3. If more than one identification is called for,keep making identifications until no more can be made.
4. End the consultation smoothly and ask the user if another consultation is required. If this is the case, another consultation is started. If not, the program stops.

Primary control of a consultation is handled by the three procedures XSHELL,XSHELL-AUX,and FINISH-XSHELL.

We begin a consultation with the query ?- xshell.

The xshell procedure has two clauses, with the first doing most of the work. It first calls xkb_intro, which supplies an introductory message for the user. The argument to xkb_intro is a list of atoms, which are displayed using the writeln procedure.

Next xshell attempts an identification. This is the core of the system, and we will need several procedures to support the interaction required to make an identification. All of these are called, indirectly, by the single goal xkb_identify (ID). We use the xkb prefix to indicate that the identification rules will be part of the XSHELL knowledge base.

In order to remember every identification that has been made, xshell uses asserta to add a fact to the definition of a special two-place predicate known. We will also use the predicate known to store other temporary information during a consultation. This is information supplied by the user or inferred from the user's information and the knowledge base. Once it has the identification,the system should inform the user of its findings. For the classification of a building, the system will respond the classification of the building in the National Building Code. The phrase " Building Identification " is stored as part of the knowledge base.

When XSHELL has made an identification, it finds this phrase, prints it, then prints the identification. Then a blank line is produced to separate the report from whatever follows.

Next XSHELL offers to explain how it reached its conclusion. After an identification has been made, reported, and explained, the system should either end the consultation or backtrack to look for other identifications. If identification is unique for the domain of the knowledge base, the consultation should end. If identification is not unique, xshell should fail and backtrack.

We tell the system whether to try to find more than one identification by putting a special fact in the knowledge base - either xkb_unique (yes) or xkb_unique (no). Then we include xkb_unique (yes) as a subgoal in xshell. If this subgoal succeds, xshell_aux is called. If xshell does not find xkb_unique (yes), it backtracks to look for another identification.

When xkb_identify (ID) fails, execution moves to the second clause in the definition of xshell, and xshell_aux is called. It simply reports that no further conclusion can be reached, and calls finish_xshell, which retracts all clauses for known and then offers to start another consultation.

3.5.1 Prop

Prop is called with the name of a property as its single

argument.

The first rule in our knowledge base is the following :

```
xkb_identify (' Group A Division 1 ') :-
                    prop (assembly_occupancy),
                    prop (performing_arts).
```

The rule states that the building is identified as ' Group A Division 1 ' if it has the properties called assembly_occupancy and performing_arts. Assembly_occupancy means that the building is used for assembly occupancies,for example, as a theatre, church etc.
Performing_arts implies a building used for the production and viewing of performing arts.
If it can be established that the subject has the property, the call to prop succeeds, otherwise it fails. In the process, one of three things happens.

1. First, prop looks in the working database to see if there is already information on the property. If there is, the call to prop succeeds or fails conclusively depending on whether the working database says the subject does or does not have the property.

2. Otherwise, prop asks the user about the property. To do this, it looks in the knowledge base for the appropriate

form of the question, then uses the yes routine to get the user's answer. If the call to yes succeeds,prop records that the subject has the property and succeeds conclusively.

3. If all attempts to establish that the subject has the property have failed, prop records that the subject does not have the property and fails conclusively.
The following routines to find out whether the building is used for assembly occupancies and for the production of performing arts, are used to get this information from the user, using the yes procedure.

```
prop (Property) :- known (Property,Value),
                   !,
                   Value == y.

prop (Property) :- xkb_question(Property,Question),
                   writeln(Question),
                   yes ('>'),nl,nl,
                   assert (known(Property,y)),
                   !.

prop (Property) :- assert (known(Property,n)),nl,nl,
                   !,
                   fail.
```

A separate question is stored in the knowledge base for each property we include in an identification rule. For example, we have the following question :

```
xkb_question (assembly_occupancy,
    [' Is the building used for assembly occupancies?',
     ' e.g,theatre,art gallery,church,club, ',
     ' court room,dance hall, day-care centre,gymnasium',
     ' lecture hall,library,licensed beverage
        establishment',
     ' lodge room,museum,passenger station,depot',
     ' recreational pier,restaurant,school, ',
     ' nonresidential undertaking premise']).
```

Starting from the first rule in the knowledge base, i.e 'Group A Division 1', Prop will ask the user if the building is used for assembly occupancies, and the user will answer yes. Then prop will ask if the building is used for the production and viewing of the performing arts. The user will answer yes, and the first rule succeeds.

Prop always looks in the database to see whether the question has already been answered before asking the question again.

3.5.2 Parameters

We define a routine called parm that asks the value of a given parameter, compares that value with the desired value, and stores the reported value for later use. Of course, parm should look to see if it already knows the answer to its question before it asks, just as prop does.

The response to parm might be a single character selected from a menu, an atom, or a number. We will create an argument place for parm where we can tell it which kind of response to expect. This argument should always be a c for a single character, an a for an atom, or an n for a number. There are three arguments to parm : parameter name, parameter type, and parameter value. The call to parm will be used as a condition in an identification rule. The call to parm succeeds if the parameter has the required value for the subject; otherwise it fails. In the process, parm does one or the other of two things :

1. Looks in the working database to see if there is already information on the parameter. If there is, the call to parm succeeds or fails conclusively depending on whether the value given in the argument matches the one stored in the database.

2. If no value is stored, parm asks the user about the parameter. The input routine used (readatom, readnumber, or a one-character response) depends on the parameter type as described by the second argument. Each type of value is

handled in a separate clause in the definition of parm. The call to parm succeeds or fails conclusively depending on whether the value given by the user matches the value given as an argument.

For example, *parm (number_streets_building_faces,c,a)* denotes that the building faces 1 street.

parm (number_streets_building_faces,c,b) implies that the building faces 2 streets.

parm (number_streets_building_faces,c,c) means that the building faces 3 streets.

The accompanying question is the following clause :

```
xkb_question (number_streets_building_faces,
    [' How many streets the building faces ?',
     ' (a) 1 street  (b) 2 streets  (c) 3 streets ']).
```

Another useful procedure is the following :

```
parmset(Parameter,Type,Set) :-
                parm(Parameter,Type,Value),
                member(Value,Set).
```

For instance, *parmset(number_storeys,c,[a,b])* means that the building has one or two storeys.

The accompanying question is the following clause :

```
xkb_question(number_storeys,
  [' What is the building height ?',
   ' (a) 1 storey',
   ' (b) 2 storeys',
   ' (c) 3 storeys',
   ' (d) 4 storeys',
   ' (e) 5 storeys',
   ' (f) 6 storeys',
   ' (g) greater than 6 storeys ']).
```

The advantage of using parmset rather a group of related properties is that the user only needs to answer a single question. This one answer is then used to satisfy or fail any number of conditions.

In the above example, a single question to the user determines the number of storeys the building has.

With a single keystroke, the user gives information that can be used to satisfy or fail several different conditions.

An XSHELL knowledge base contains a set of identification rules, each of them a clause for the xkb_identify predicate. Each rule uses the predicates prop, parm, parmset in its conditions or subgoals. For every property or parameter named in a condition for an identification rule, we will also need to include an appropriate question in the

knowledge base. These questions will be stored in clauses for the predicate xkb_question.

3.5.3 Explanation

The explanation facility displays the rule used to derive the last conclusion reported. However, it does not tell the user how XSHELL satisfies the conditions for this rule. The explanatory facility can be turned on and off. If xkb_explain(no) is in the knowledge base, XSHELL will not explain its conclusions. If explanations are required, xkb_explain(yes) should be inserted in the knowledge base. When explain is called, it takes three successive courses of action until one succeeds.

1. It looks to see if xkb_explain(no) is in the knowledge base. If so, no explanation is required, so explain calls the auxiliary procedure wait and succeeds conclusively.
2. If (1) fails, then an explanation is offered, so explain asks the user if an explanation is wanted. If the answer is no, explain succeeds conclusively.
3. If (2) also fails, the user wants an explanation, so explain looks in the working database to find the last identification that was remembered, finds an identification rule that supports this identification, and tests to see if the condition for the identification rule succeeds. Since the identification was made, some rule will finally qualify. This rule is displayed and explain succeeds. The rule which

is printed will contain predicates prop,parm,parmset which may not be helpful to the user. The real value of the explanatory facility in XSHELL is as a tool the knowledge base builder can use to debug knowledge bases.

3.5.4 Applications

XSHELL is the main program or procedure for the expert system consultation driver. Xshell is the procedure to identify the building as specified in the National Building Code.

xshell1 is the procedure to extract the fire regulations related to a particular building.

In the National Building Code, fire regulations depend on (a) identification of the building (b) number of storeys (c) building area (d) whether the building is sprinklered or not.

The values of the building area used in the clauses for fire regulations are grouped together in the parameters area_unsprinklered_Group_X and area_sprinklered_Group_X, where X stands for A, B, C, D, E, or F.

The predicate parmset (area_unsprinklered_Group_X,c,[ ]) facilitates the writing of rules in the knowledge base.

Xshell1 asks the following question :

```
xkb_question (area_unsprinklered_Group_A,
 [' What is the building area ?',
 ' choose the closest number greater than the building
   area',
 '',
 '(a) 400 m2 (b) 500 m2 (c) 600 m2 (d) 800 m2 (e) 1000 m2',
 '(f) 1200 m2 (g) 1250 m2 (h) 1500 m2 (j) 1600 m2 (k) 1800
      m2',
 '(l) 2000 m2 (m) 2400 m2 (n) 2500 m2 (p) 3000 m2 (q) 3600
      m2',
'(r) 4000 m2 (s) 5000 m2 (t) 6000 m2 (u) greater than 6000
m2']).
```

When the user gives the value of the building area, the fire regulation goal is achieved by satisfying one of the rules in the knowledge base.

For example, clause 3.2.2.24 in the National Building Code states the following :

(1) A building classified as Group A, Division 3 shall conform to Sentence (2) provided the building

(a) is not more than 1 storey in building height,

(b) if unsprinklered, has a building area not more than

(i) 1000 m2 if facing 1 street,

(ii) 1250 m2 if facing 2 streets, or

(iii) 1500 m2 if facing 3 streets, and

(c) if sprinklered, is not more than twice the area limits of clause (b).

(2) The building shall be of combustible or noncombustible construction used either singly or in combination.

This clause was written as Rule 20 :

```
xkb_fireregulation (
['Building shall be of combustible or
  noncombustible  construction',
 ' used either singly or in combination.']) :-
  xkb_identify(' Group A Division 3 '),
  parmset(number_storeys,c,[a]),
(( prop(unsprinklered),
((parm(number_streets_building_faces,c,a),
  parmset(area_unsprinklered_Group_A,c,[a,b,c,d,e]))
| (parm(number_streets_building_faces,c,b),
```

```
   parmset(area_unsprinklered_Group_A,c,[a,b,c,d,e,f,g]))
 | (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_A,c,[a,b,c,d,e,f,g,h])))))
 | ((\+ prop(unsprinklered)),
  ((parm(number_streets_building_faces,c,a),
    parmset(area_sprinklered_Group_A,c,[a,b,c,d,e]))
 | (parm(number_streets_building_faces,c,b),
    parmset(area_sprinklered_Group_A,c,[a,b,c,d,e,f,g]))
 | (parm(number_streets_building_faces,c,c),
parmset(area_sprinklered_Group_A,c,[a,b,c,d,e,f,g,h])))))).
```

xshell2 is the consultation driver to determine the fire alarms required for a building.

xshell3 is the procedure to explain the meaning of the terminology used in the clauses in the National Building Code.

xshell4 deals with the safety requirement clauses.

xshell5 is the procedure to determine exit requirements.

xshell6 deals with the plumbing facilities.

XSHELL is a flexible expert system shell that can be easily modified by the user. For example, future requirements in the National Building Code can be incorporated in XSHELL, by writing a new procedure xshell7, independently of the existing procedures xshell1 to xshell6.

3.6 MENU

The file *menu.pl* contains the procedure *menu*, which generates menus automatically. Given a list of choices, menu displays them in a numbered list, insists that the user type a valid number, and then returns the value specified for that choice.

A menu is described by a list of structures such as :

```
[ item ( Choice 1, Value 1),
      item ( Choice 2, Value 2 ),
       item ( Choice 3, Value 3 ) ....]
```

where *Choice 1* is a message describing one of the items to choose from, *Value 1* is the result to be returned by the menu routine if the user chooses that item, and so forth.

Procedure *menu* works as follows :

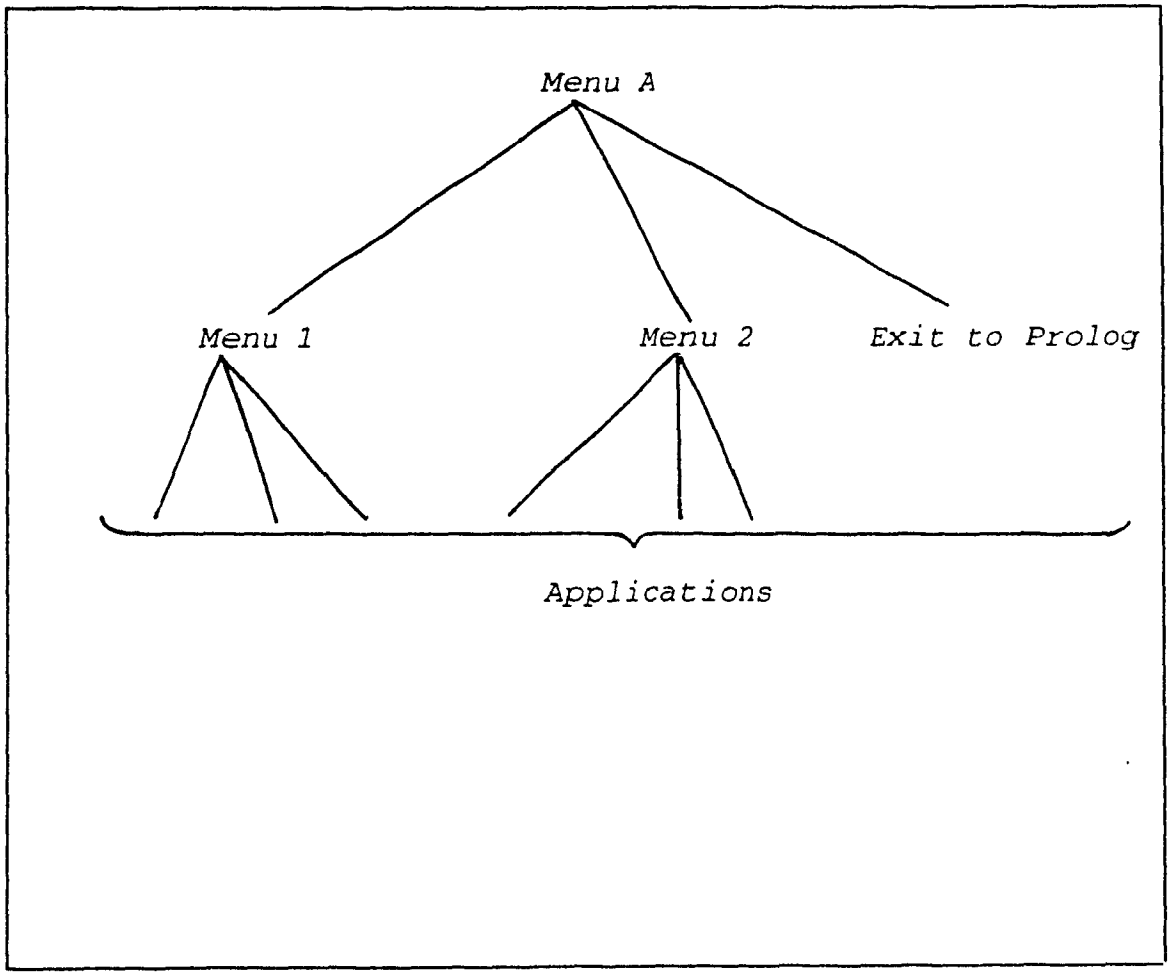
1. Subroutine *menu_display* works through the list of items, displaying the choices and simultaneously counting them. The count begins with 49 (the ASCII code for the integer "1"). At the end, *Last* is instantiated to the ASCII code for the last digit used.
2. Subroutine *menu_choose* accepts a digit from the user, and *menu_choose_aux* checks its validity.
3. Subroutine *menu_select* works through the list of items again, counting them until it finds the item corresponding to the digit typed.

It is to be noted that the value returned can be any Prolog term. Thus ' menu 1 ' will display a menu of routines and then execute one of them :

```
Menu 1
1   Building Identification
2   Fire Regulation
3   Fire Alarm
4   Safety Requirements within floor areas
5   Requirements for exits
6   Plumbing facilities
7   Term Definition
8   Previous menu
    Your choice (1 to 8) :
```

3.6.1 Tree-structured Menu Systems

Systems of menus usually have a tree structure : there is a main menu, and under it a set of subordinate menus, and under these a set of yet lower-level menus, and so forth.



The following are the things one might want to do from a given menu.

1. Enter an application (i.e, run a routine other than a menu).
2. Choose a subordinate menu and enter it.
3. Go back to the previous menu.
4. Jump to a menu somewhere else in the tree.

In the file *MENUSYST.PL*, each menu knows the path by which the user reached it, so that it will be possible to back up. Each menu is expressed as a procedure, and the path is passed to it as an argument, in the form of a list of procedure names.

When the user goes from one menu to another, the current menu adds its own name to the beginning of the path and passes the result to the new menu. The path is therefore a list of all the menus the user has been through, with the most recent one first. Backing up is accomplished by calling the menu named by the first element of the path, with the remainder of the path as its argument.

MENUSYST.PL has seven application programs that can be called from menus. A session with our sample system looks like this :

```
Main menu (menu A)
1 Building Regulations
2 Query with constraints
3 Exit to unix
4 Exit to prolog
 Your choice (1 to 4) :
```

```
Menu 1
 1 Identify the building
 2 Fire Regulation
 3 Fire Alarm
 4 Safety Requirements within floor areas
 5 Requirements for exits
 6 Plumbing facilities
 7 Term Definition
 8 Previous menu
 Your choice (1 to 8) :
```

```
Menu 2
1 Fire Regulation
2 Fire Alarm
3 Safety Requirements within floor areas
4 Requirements for exits
```

```
5 Plumbing facilities
6 Previous menu
Your choice (1 to 6) :
```

Here "previous menu" means the menu that we actually saw in the previous step, not the menu directly above the current one in the tree structure. All this is implemented by means of three special procedures : *forward*, for proceeding to another menu; *back*, for backing up along the current path; and *application*, which calls an application and then returns to the current menu.

3.6.2 WRITELN

The procedure *writeln* displays single - or multi-line messages. Given a list of atoms, it displays the atoms in sequence, each on a separate line. If the argument of *writeln* is a list that contains other lists, only the top-level list is taken apart; the inner lists are displayed as lists.

We usually represent messages as atoms ('hello there') rather than strings ("hello there") because atoms occupy less space in memory. Atoms are stored as contiguous sequences of characters. However, strings are represented as linked lists of ASCII codes. This means that each character occupies the space needed for an ordinary integer (two bytes

rather than one).

Atoms do have one disadvantage. Once created, they are kept in an object list whether or not they are still in use.

Much of the input that a program accepts from a user consists of answers to yes-no questions. The listing yes-no contains a procedure that displays a question, insists that the user type *Y or N* in response, and then succeeds if the user typed *Y*, or fails if he typed *N*.

3.6.3 Input of Strings and Atoms

The procedure *readstring* enables the user to type a string without following Prolog syntax. All of the characters, except the final Return, typed by the user, are returned as a list of ASCII codes.

The algorithm is as follows :

1. *Read one character (char).*
2. *Produce a list of all characters (Result) by recursion.*
3. *The answer is [char|Result].*

Readstring also allows for backspacing. The character that is being erased must disappear from the screen. This is accomplished in the following manner : when the user hits Backspace, the cursor moves one space to the left, so that

it is placed on the character to be erased. The current cursor position is blanked out by outputting a blank. Then, the erased character is removed from the string as follows : if the list of subsequent characters begins with a backspace, then both the backspace and the characters in front of the backspace, should be discarded.
Assume that C1,C2, and C3 stand for characters other than backspaces (ASCII code = 8).
The procedure *readstring_cons* achieves the following effects:

C1 plus [C2,C3...] gives [C1,C2,C3...]' but
C1 plus [8,C2,C3...] yields [C2,C3,...].
Also, 8 plus [8,C2,C3...] gives [8,8,C2,C3,...].

The file *READINT.PL* defines a procedure called *readinteger* that accepts integers from the keyboard. A string is read using the procedure *readstring* and it is then converted into an integer. The conversion is achieved with the help of the variable *SoFar* that contains the number formed by the characters already converted. Thus the string "123" is interpreted first as 1, then as 12, then as 123.
SoFar is initialized to 0.
The algorithm for the conversion is as follows :

1. If the character is a blank, ignore it.
2. If the character is a digit, then multiply SoFar by 10

and add the numeric value of this digit.

3. *If the string is empty, Result equals SoFar and terminate.*
4. *If none of these cases applies, then the user must have typed a non-numeric character, so ask him to retype the whole number.*

CHAPTER 4

LOGICAL INFERENCE MECHANISM

4.1 Needs for a logical inference mechanism

As discussed earlier, the traditional forward chaining mechanism for expert systems is useful when the user knows all the physical constraints of the building. Previous implementations of expert systems by some other groups have used the forward chaining mechanism. For example, Rasdorf and Wang [15] used the forward chaining mechanism for the expert system implementation of BOCA (Building Officials and Code Administrators International) building code, which we shall discuss later in chapter 5. Traditional forward chaining mechanism asks several yes-no questions to the user. The user answers according to his knowledge about the physical constraints of the building. According to these answers, the system identifies suitable legal clauses, as we implement for menu 1. However, assume that the user has a partial knowledge of the physical constraints of the building. In this case, we cannot follow the forward chaining mechanism. Therefore, the system cannot find the appropriate legal clause. We solve this problem using the logical inference engine based on Prolog's resolution mechanism. For example, the user wants to build an arena and he wishes to find the fire regulations in the building code.

He specifies his constraints such as the location, say a corner of two streets, and the type of building, i.e, an arena. Under these user-defined physical constraints, the logical inference engine displays the fire regulations in the building code, as well as the conditions required to satisfy the code provisions. Or, when the user does not want to install sprinklers for some reason (say a budget reason, or some local difficulty of water system) and the user wants to build an arena, he wishes to know in what kind of location and how many storeys he can build. In that case, he puts another constraint on the sprinklers, say " no sprinkler ". Then the same logical inference engine can solve this query under the new constraints. In general, there might be no solution for a certain condition in the building code. So, the user needs to know all possible conditions which have solutions as well as the solution itself. In other words, the logical inference mechanism shows which conditions are possible and which conditions are impossible. For example, the code provision 3.2.2.20 applies for a sprinklered assembly building Division 2. By using the logical inference mechanism, the user will find that the solution calls for a building which must be sprinklered. This example shows the advantage of symbolic manipulation for the condition by using variables instead of values. Besides, it shows the advantage of using a logic programming language instead of a procedural language for query-related

implementation of legal codes.

4.2 Implementation

4.2.1 Introduction

Now, we shall explain the implementation of Menu 2, which deals with the logical inference mechanism. Menu 2 has the following format :

```
Menu 2
1  Fire Regulation
2  Fire Alarm
3  Safety Requirements within floor areas
4  Requirements for exits
5  Plumbing facilities
6  Previous menu
Your choice (1 to 6) :
```

The user chooses menu 2 when he wishes to define his constraints. With our present system, he has the choice of item 1 to item 5 in menu 2. A customized user interface has been written in Prolog. The program source code is given in the file query.pl on page clxxiii of the Appendix. We use Prolog's resolution mechanism and treat the rule number as a variable, as well as other constraints as variables. The user does not know the rule number and it is up to the

system to find the rule number. Now, we describe briefly the logical inference mechanism.

4.2.2 The resolution mechanism

The rule of resolution is a rule of inference, i.e, it tells us how one proposition can follow from others. The Prolog system is based on a resolution theorem prover for Horn clauses. The particular strategy that it uses is a form of linear input resolution. We start with the goal statement and resolve it with one of the hypotheses to give a new clause. Then we resolve that with one of the hypotheses to give another new clause, and so on. The rule of resolution forms the basis of the Prolog inference mechanism :

```
  ~ A V B
  ~ B V  C        Resolution
----------------
  ~ A V C
```

The logical connective *or* is written as V in logic. The connective ~ means not. The above rule shows that given the two statements ~ A V B and ~ B V C, a new statement ~ A V C can be concluded from the two statements. A prolog query *p* asks the system to prove the statement *p* from a set of statements {a,b,....} - the clauses contained in the knowledge base. Prolog proves the statement *p* by proving that {a,b,.....,~ p} is contradictory, i.e, that the empty

clause {} can be deduced from it.

4.2.3 Use of logical variables and metapredicates

Our knowledge base for fire regulations for logical inference mechanism has the following format :

```
rule(A,S,X,Y,Z) :-
                    .
                    .
                    X = 2,
                    Y = 1000,
                    .
                    .
```

The variable *A* denotes the rule number, *X* the number of storeys, and *Y* the building area in square meters.
When the user queries the sysetm with the statement rule(A,S,X,Y,Z), the system finds a clause whose head matches one of the goals, instantiate variables as necessary, remove the goal that matches, and then add the body of the instantiated clause to the goals to be satisfied. For example, the variable X is instantiated to the value 2.
The range of values for the building area is characteristic of the building code. We have two cases to interpret one

numerical value in the building code according to the different types of query. First, the building area is given an exact number by the user. In this case, ths system needs to recognize which range it belongs to. For instance, referring to Example 1 on page 16 of the legal clause in the code, clause 3.2.2.25 applies if the user defines the building area as say 500 m2. The program source code is given on page xvii of the Appendix. Second, the user does not know the building area, and he wishes to know the maximum area in m2 under some other known constraints by the user. In this case, the system answers by displaying the maximum building area in m2. We implement both cases of the building area by using Quintus Prolog metapredicates **var** and **nonvar**. The goal ***var****(X)* succeeds if *X* is currently an uninstantiated variable. The goal ***nonvar****(X)* succeeds if *X* is not currently an uninstantiated variable. The predicate ***nonvar*** is therefore the opposite of ***var***. The use of metapredicate solves the problem which is common in legal codes involving numerical values. However, figure-related problems should be treated with special attention. In our case, we have to implement the case when the building area is a variable, and the case when the building area is given as an integer.

4.2.4 The main procedure

The litteral find_solutions (*Q*) for a query *Q*, is shown on

page clxxvii of the Appendix. Find_solutions (*Q*) calls the query *Q*, and then displays *Q* with instantiations. Then find_solutions (*Q*) asks if more solutions are wanted. If the user answers " yes ", the system finds another solution if there is one. If there is no solution, the system answers with the statement " No (more) solutions ". If the user types N or n (for "no"), find_solutions (*Q*) executes a cut, and the system displays the main query menu 2. For each item in menu 2, find_solutions (*Q*) extracts the corresponding variables from the user-input query using the metapredicate *arg(A,Q,B)*. The latter unifies *B* with the *A*th argument of structure *Q*. The customized user interface file query.pl makes Prolog much more user-friendly. The menu displays the solutions explicitly in an English format, which helps to enhance better comprehension of the output messages. The litteral *write_heading* in the file *query.pl* outputs the solutions to a query. The metapredicates *var* and *nonvar* are used in the processing of the values of the variables in the solutions. If the variable in the solution is instantiated, its value is displayed. If the variable in the solution is uninstantiated, the system displays " undefined ".

4.2.5 Correspondence between the legal codes and the Horn clauses

The correspondence between the legal codes and the Horn clauses was described in section 2.3. The conditions of a

provision for the fire regulations in the building code are represented by tables such as Table 3.2.2.B shown on page 16. Table 3.2.2.B has been implemented as rule 27 in the program :

```
rule(27,S,X,Y,Z) :-
                    xkb_identify(' Group A Division 3'),
                    (( X = 1,
     ((S = 1,
          (((Z = 1,var(Y),Y = 4000)
            | (Z =1,nonvar(Y),Y =< 4000,Y > 2400))
          | ((Z = 2,var(Y),Y = 5000)
             | (Z = 2,nonvar(Y),Y =< 5000,Y > 3000))
          | ((Z = 3,var(Y),Y = 2400)
             | (Z = 3,nonvar(Y),Y =< 6000,Y > 3600))))
        | (S = 2,
            (((Z = 1,var(Y),Y = 8000)
              | (Z = 1,nonvar(Y),Y =< 8000,Y > 4800))
            | ((Z = 2,var(Y), Y = 10000)
               | (Z = 2,nonvar(Y),Y =< 10000,Y > 6000))
            | ((Z = 3,var(Y),Y = 12000)
                | (Z = 3,nonvar(Y),Y =< 12000,Y > 7200))))))
                   (( X = 2,
     ((S = 1,
          (((Z = 1,var(Y),Y = 2000)
```

```
      | (Z =1,nonvar(Y),Y =< 2000))
    | ((Z = 2,var(Y),Y = 2500)
      | (Z = 2,nonvar(Y),Y =< 2500))
    | ((Z = 3,var(Y),Y = 3000)
      | (Z = 3,nonvar(Y),Y =< 3000))))
  | (S = 2,
      (((Z = 1,var(Y),Y = 4000)
        | (Z = 1,nonvar(Y),Y =< 4000))
      | ((Z = 2,var(Y), Y = 5000)
        | (Z = 2,nonvar(Y),Y =< 5000))
      | ((Z = 3,var(Y),Y = 6000)
        | (Z = 3,nonvar(Y),Y =< 6000))))))).
```

We described earlier the use of metapredicates **var** and **nonvar**. The metapredicate **var** is used for the case when the user does not know the building area in square meters. The system will output the required maximum building area in the building code. The metapredicate **nonvar** is used to handle the case when the user defines his own constraint for the building area. In this case, the building area should satisfy a range of numerical values in order to conform to the requirements of the building code. Sometimes, the condition part of the provisions in the building code contains an exception clause. In this case, a new rule is written to handle the exception part. For example, the

exception part in provision 3.2.2.25 in the code is written as rule 25 :

```
rule(25,S,X,Y,Z) :-
                    xkb_identify(' Group A Division 3'),
                    (( X = 1,
     ((S = 1,
          (((Z = 1,var(Y),Y = 1200)
            | (Z =1,nonvar(Y),Y =< 1200))
          | ((Z = 2,var(Y),Y = 1500)
             | (Z = 2,nonvar(Y),Y =< 1500))
          | ((Z = 3,var(Y),Y = 1800)
             | (Z = 3,nonvar(Y),Y =< 1800))))
        | (S = 2,
            (((Z = 1,var(Y),Y = 2400)
              | (Z = 1,nonvar(Y),Y =< 2400))
            | ((Z = 2,var(Y), Y = 3000)
              | (Z = 2,nonvar(Y),Y =< 3000))
            | ((Z = 3,var(Y),Y = 36000)
               | (Z = 3,nonvar(Y),Y =< 36000))))).
```

4.3 Examples

The logical inference engine works backward from hypothesized consequents to locate known predicates that would provide support. When there are many solutions to a

goal, and a list of all those solutions is desired, we can use Prolog built-in predicates **bagof** and **setof**. Bagof(X,P,B) means that the bag of instances of X such that P is provable is B.

setof(X,P,S) is defined as follows :
S is the set of instances of X such that P is provable.

In the procedure " Query ", as shown in the program listings, the variable X denotes the number of storeys of the building; the variable Y represents the area of the building.

A = rule number

S = status whether the building is sprinklered or not

S = 1 (unsprinklered)

S = 2 (sprinklered)

For example, given a building area of 4000 square meters,and the building is sprinklered, the user can extract from the system, the possible values for the number of storeys.
A sample dialogue using the query menu is as follcws :

Query

Rules are defined as rule(A,S,X,Y,Z)

Use the following variables

```
X = number of storeys
Y = building area in m2
Z = number of streets the building faces
A = rule number
S = status whether the building is sprinklered or not
S = 1  (unsprinklered)
S = 2  (sprinklered)
: bagof(X,rule(A,2,X,4000,2),L).
```

The system responds with the following yes-no questions to find the building classification.

```
Is the building used for assembly occupancies?
e.g,theatre,art gallery,church,club,court room,
dance hall,day-care centre,gymnasium,lecture hall
library,licensed beverage establishment,lodge room
museum,passenger station,depot,recreational pier,
restaurant,school,nonresidential undertaking premise
> y


Is the building used for the production and viewing
of the performing arts?
> n
```

Is the building used as an arena?

e.g,ice rink, indoor swimming pool

> **y**

Is the building used for the congregation or gathering of persons participating in or viewing open air activities?

e.g,amusement park,bleach,grandstand,stadium.

> **n**

Solution found: *bagof(_1109,rule(24,2,_1109,4000,2),[1])*

Number of storeys = 1

Fire regulation

rule 24 (provision 3.2.2.25)

Building shall be of combustible or noncombustible construction used singly or in combination, and the fire resistance rating for

(a) mezzanines > 45 min

(b) loadbearing walls,columns and arches > 45 min

or shall be of noncombustible construction.

Look for another? (Y/N) :**y**

Solution found: *bagof(_1321,rule(26,2,_1321,4000,2),[1])*

Number of storevs = 1

Fire regulation

rule 26 (provision 3.2.2.25)

Fire resistance rating, if of combustible construction for roof assemblies *> 45 min*

Look for another? (Y/N) : y

Solution found: bagof(_1321,rule(27,2,_1321,4000,2),[2])

Number of storeys = 2

Fire regulation

rule 27 (provision 3.2.2.26)

Building shall be of noncombustible construction, and the fire resistance rating for

(a) floor assemblies (fire separations) *> 1 h*

(b) mezzanines *> 1 h*

(c) roof assemblies *> 45 min*

or be of heavy timber construction,and

(d) loadbearing walls,columns and arches *> that required for the supported assembly*

*Look for another? (Y/N): **y***

No (more) solutions

This sample session shows that the system has found the answer to the query; the maximum number of storeys is 2. The second example shows that the system will supply the maximum building area according to the code, given the facts that the building has 3 storeys, it is sprinklered, and it faces 2 streets.

The query is as follows :

bagof(Y,rule(A,2,3,Y,2),L).

Is the building used for assembly occupancies?
e.g,theatre,art gallery,church,club,court room,
dance hall,day-care centre,gymnasium,lecture hall
library,licensed beverage establishment,lodge room
museum,passenger station,depot,recreational pier
restaurant,school,nonresidential undertaking premise.
> **n**
Is the building used as an institution?
e.g,jail,penitentiary,police station with detention
quarters,prison,hospital,reformatory,convalescent home,
home for the aged,nursing home,orphanage,sanitorium.
> **n**
Is the building used for sleeping accommodations

```
excluding institutions?
> y
```

```
Solution found : bagof(_2502,rule(35,2,3,_2502,2),[1500])
Building area = 1500 m2

Fire regulation
rule 35 (provision 3.2.2.34)
Building shall be of noncombustible construction, and
the fire resistance rating for
(a) floor assemblies                          > 45 min
(b) mezzanines                                > 45 min
(c) loadbearing walls,columns          > that required for
    and arches                           supported assembly

Look for another?  (Y/N) : y
```

```
Solution found : bagof(_2502,rule(36,2,3,_2502,2),[1500])
Building area = 1500 m2

Fire regulation
rule 36 (provision 3.2.2.34)
```

Floor assemblies, including floors over basements, which are entirely contained within such dwelling units, shall have a fire resistance rating > 45 min and need not be constructed as fire separations.

Look for another? (Y/N) : y

Solution found : bagof(_2502,rule(38,2,3,_2502,2),[2000])

Building area = 2000 m2

Fire regulation

rule 38 (provision 3.2.2.35)

Building shall be of combustible or noncombustible construction used singly or in combination,and the fire resistance rating for

(a) floor assemblies		*> 1 h*
(b) mezzanines		*> 1 h*
(c) roof assemblies		*> 1 h*
(d) loadbearing walls,columns and arches	*> that required for supported assembly*	

Look for another? (Y/N) : **y**

Solution found : bagof(_2502,rule(39,2,3,_2502,2),[2000])

Building area = 2000 m2

Fire regulation

rule 39 (provision 3.2.2.35)

Floor assemblies, including floors over basements, which are entirely contained within such dwelling units, shall have a fire resistance rating > 1 h and need not be constructed as fire separations.

Look for another? (Y/N) : y

Solution found : bagof(_2502,rule(44,2,3,_2502,2),[10000])

Building area = 10000 m2

Fire regulation

rule 44 (provision 3.2.2.37)

Building shall be of noncombustible construction, and the fire resistance rating for

(a) floor assemblies (fire separations)	*> 1 h*
(b) mezzanines	*> 1 h*
(c) roof assemblies	*> 1 h*
(d) loadbearing walls,columns and arches	*> that required for supported assembly*

Look for another? (Y/N) : **y**

Solution found : bagof(_2502,rule(45,2,3,_2502,2),[10000])

Building area = 10000 m2

Fire regulation

rule 45 (provision 3.2.2.37)

Floor assemblies, including floors over basements, which are entirely contained within such dwelling units, shall have a fire resistance rating > 1 h and need not be constructed as fire separations.

Look for another? (Y/N) : y

Solution found :

bagof(_2502,rule(46,2,3,_2502,2),[unlimited])

Building area = unlimited m2

Fire regulation

rule 46 (provision 3.2.2.38)

Building shall be of noncombustible construction, and the fire resistance rating for

(a) floor assemblies (fire separations) > 2 h

(b) mezzanines > 1 h

(c) roof assemblies > 1 h

(d) loadbearing walls,columns > that required for

and arches *supported assembly*

Look for another? (Y/N) : **y**

Solution found :

bagof(_2502,rule(47,2,3,_2502,2),[unlimited])

Building area = unlimited m2

Fire regulation

rule 47 (provision 3.2.2.38)

Floor assemblies, including floors over basements, which are entirely contained within such dwelling units, shall have a fire resistance rating > 1 h and need not be constructed as fire separations.

Look for another? (Y/N) : y

No (more) solutions

The third example shows that the empirical relacionships between the number of storeys and the building area, can be extracted from the system, by asking the following query: ***rule(A,2,X,Y,2).***

Is the building used for assembly occupancies?

e.g, theatre, art gallery, church, club, court room,
dance hall, day-care centre, gymnasium, lecture hall
library, licensed beverage establishment, lodge room
museum, passenger station, depot, recreational pier
restaurant, school, nonresidential undertaking premise.

> **n**

Is the building used as an institution?
e.g, jail, penitentiary, police station with detention
quarters, prison, hospital, reformatory, convalescent home,
home for the aged, nursing home, orphanage, sanitorium.

> **n**

Is the building used for sleeping accommodations
excluding institutions?

> **n**

Is the building used for conducting business and
rendering of professional and personal services?

> **n**

Is the building used for displaying, or selling of retail
goods, wares or merchandise?
e.g, department store, market, shop, supermarket

> **y**

Solution found : rule(53,2,1,2500,2)

Number of storeys = 1

Building area = 2500 m2 max

Building is sprinklered

Building faces 2 streets

Fire regulation

rule 53 (provision 3.2.2.43)

Building shall be of combustible or noncombustible construction used singly or in combination, and the fire resistance rating for

(a) floor assemblies (fire separations)		*> 45 min*
(b) loadbearing walls, columns and arches		*> that required for supported assembly*

Look for another? (Y/N) : **y**

Solution found : rule(54,2,1,6000,2)

Number of storeys = 1

Building area = 6000 m2 max

Building is sprinklered

Building faces 2 streets

Fire regulation

rule 54 (provision 3.2.2.44)

Building shall be of combustible or noncombustible

construction used singly or in combination,and

the fire resistance rating for

(a) floor assemblies (fire separations) > 45 min

(b) mezzanines > 45 min

(c) roof assemblies > 45 min

(d) loadbearing walls,columns and arches > 45 min

or shall be of noncombustible construction, except that such members and assemblies supporting a fire separation shall have a fire resistance greater than that required for the supported assembly.

Look for another? (Y/N) : y

Solution found : rule(54,2,2,3000,2)

Number of storeys = 2

Building area = 3000 m2 max

Building is sprinklered

Building faces 2 streets

Fire regulation

rule 54 (provision 3.2.2.44)

Building shall be of combustible or noncombustible construction used singly or in combination,and the fire resistance rating for

(a) floor assemblies (fire separations) > 45 min

(b) mezzanines > 45 min

(c) roof assemblies > 45 min

(d) loadbearing walls,columns and arches > 45 min

or shall be of noncombustible construction, except that such members and assemblies supporting a fire separation shall have a fire resistance greater than that required for the supported assembly.

Look for another? (Y/N) : y

Solution found : rule(54,2,3,2000,2)

Number of storeys = 3

Building area = 2000 m2 max

Building is sprinklered

Building faces 2 streets

Fire regulation

rule 54 (provision 3.2.2.44)

Building shall be of combustible or noncombustible construction used singly or in combination,and the fire resistance rating for

(a) floor assemblies (fire separations) > 45 min

(b) mezzanines > 45 min

(c) roof assemblies > 45 min

(d) loadbearing walls,columns and arches > 45 min

or shall be of noncombustible construction, except that such members and assemblies supporting a fire separation shall have a fire resistance greater than that required for the supported assembly.

Look for another? (Y/N) : y

Solution found : rule(56,2,1,unlimited,2)

Number of storeys = 1

Building area = unlimited m2 max

Building is sprinklered

Building faces 2 streets

Fire regulation

rule 56 (provision 3.2.2.45)

Building shall be of noncombustible construction, and the fire resistance rating for

(a) floor assemblies (fire separations)	*> 2 h*	
(b) mezzanines	*> 1 h*	
(c) roof assemblies	*> 1 h*	
(d) loadbearing walls,columns and arches	*> that required for supported assembly*	

Look for another? (Y/N) :

.
.
.
No (more) solutions

The statement " No more solutions " is meaningful in some cases. For example, the user wishes to find the fire regulations and he uses menu 2. The system responds with the fire regulation provisions as well as the conditions for the number of streets the building faces. In this case, the maximum number of streets the building can face according to the code, is " 3 ". The system responds with " No more solutions " after the condition for the maximum number of streets the building faces, has been output. The statement " No more solutions " is meaningful in this case.

CHAPTER 5

COMPARISON WITH OTHER APPROACHES TO CODE REPRESENTATION

We compare our prototype system with other previous approaches to knowledge representation used for processing legal codes. We classify the traditional methods of knowledge representation related to legal codes. Legal codes are generally modeled by the following schemes: decision tables, production systems, frames, semantic networks, and predicate logic based on Prolog.

5.1 Decision table

A decision table represents antecedent-consequent relationships among variables or datums. Rasdorf and Wang [15] have reviewed the use of decision tables for processing design codes. Fig. 3 shows a decision table corresponding to clause 4.1.7.1 of the National Building Code. The upper-left portion of the table contains a set of conditions for instantiated values for the datums. The upper-right part shows whether a condition is satisfied or not. The lower-left part contains a set of actions to be taken. The lower-right portion shows which action is to be taken if a set of conditions are satisfied in a given column.

c_s = slope factor to compute loading due to snow accumulation on a roof

conditions	**rules**		
roof slope	1	2	3
$\alpha \leq 30°$	Y	N	N
$30° \leq \alpha \leq 70°$	N	Y	N
$\alpha > 70°$	N	N	Y
actions			
$c_s = 1.0$	X		
$c_s = \frac{70° - \alpha}{40°}$		X	
$c_s = 0$			X

Fig. 3 Decision Table for clause 4.1.7.1 of Building Code

5.2 Production systems

Production systems are rule-based systems containing condition-action rules called productions.

A rule has the following format:

if <antecedent> **then** <consequent>

where antecedent is a set of conditions:

<condition 1>,<condition 2>,....

consequent is a set of actions :

<action 1>,<action 2>,.....

Production systems are a natural way of representing knowledge. They have the advantage of being modular, i.e, rules are independent of each other. New rules may be added or deleted independently of other rules. Rasdorf and Wang [15] proposed a production system environment for processing standards and codes. Their approach was to convert the decision tables into production rules, and their system was written in the OPS5 knowledge engineering language [16]. Their system performs design conformance checking and provides allowable value ranges for undetermined data. The term " generic " is used in the sense that the system is independent of a particular design standard or code. Their system converts only once the code provisions into an internal format. Existing standards can be updated by simply revising the applicable part of the code in the knowledge base. Besides, the designer can arbitrarily choose those portions of the code he is

interested in. In Rasdorf and Wang's system, two standards processing knowledge-based expert systems have been implemented : *Query Monitor*, which involves data retrieval from a code, and *Roofload Checker*, which performs design conformance checking utilizing a standard. The *Query Monitor* was developed using the commercial development tool M.1. The knowledge representation scheme of M.1 consists of production rules. The *Roofload Checker* was written in the OPS5 knowledge engineering language. Their system partitions the facts into independent modules with the advantage that changes to one module do not affect the others.

Standards are represented as provisional and organizational facts. A provisional fact corresponds to the combination of a datum and its associated decision table condition entry. For example, " c_3 = 1.0 " is a provisional fact. An organizational fact represents the hierarchical relationships among the datums of a standard. The generic knowledge-based standards processing architecture was implemented in OPS5 with a BLISS-based compiler installed on a VAXSTATION-II running the VMS operating system.

Provisional facts are represented by the class cond:

```
(literalize cond
            item
            provid
            col
            oprl
```

val1

opr2

val2)

where item records the name of a datum; provid refers to which provision the datum appears in; col is the column number in the governing decision table;

opr1 records the relational operator =, >, or ≥;

opr2 records the relational operator < or ≤;

val1, val2 represents data item values.

It can be observed that Rasdorf's SPIKE architecture involves a lot of processing and programming efforts. As a result, it takes more time to develop the expert system, and verify the system. Each provisional fact has to be declared as in a procedural language. In OPS5, all objects and attributes must be declared before their use in a rule. This declaration is done with a "literalize" statement. In Rasdorf and Wang 's system, all the clauses in the code have to be converted in the format of decision tables.

Rasdorf's system is similar to our Prolog-based knowledge representation in the sense that Prolog is based on facts and rules. The main difference from our system is that the knowledge engineer using Rasdorf's system, must do a lot of processing to convert the decision tables into production rules. The main feature of our system is the capability of the user to extract from the system, all possible relationships between variables based on user-defined

constraints. This type of query is not available in Rasdorf and Wang's system. The reason is that OPS5 lacks the *resolution rule* inbuilt in Prolog. The inference engine of Prolog is based on the following two rules :

(a) the *resolution rule* which determines particular facts from other known facts, and provides an automatic way of proving theorems from axioms. Using this rule, one can establish the fact *b* from a database of facts *a* and *a -> b*.

(b) the *unification rule* which matches variables in predicate formulae. For example, if there is a fact *parm(number_storeys,c,2)* and a goal *parm(number_storeys,c,X)*, where X is a variable, then the *unification rule* matches X to 2.

The inference engine for OPS5 is very simple, and is termed as *"recognize-act" cycle*. Rules are compared to the elements in working memory until a rule fires and new information is placed in working memory. The *recognize-act cycle* continues until the elements in working memory do not satisfy the condition part of any production rules.

We have shown in chapter 1, the example where the user can query our system to find out all possible areas of his building related to the fire-resistance of the building structural components, under the constraints that the building is unsprinklered and faces 2 streets. The inference engine in OPS5 cannot handle this type of query. Besides, it

is difficult to combine the language OPS5 with a logic programming language such as Prolog.

5.3 Frames

Frames or schema have a name, a number of slots, and each slot may have a number of facets. A slot in a schema may be an attribute or a relation. A relation slot is used to link two frames together.

Dym, Henchey, Delis and Gonick [17] have developed a knowledge-based expert system capable of reviewing an architectural design for conformance with the Life Safety Code (LSC) of the National Fire Protection Association. Their LSC advisor was developed in conjunction with Graphic Horizons Inc's Graph/Net computer-aided design and drafting (CADD) system. The protoytpe LSC Advisor was implemented in Intellicorp's Knowledge Engineering Environment (**KEE**), using the Texas Instruments Explorer 1 workstation. Their system reviewed the following design procedures:

(1) identification of the parts of the building
(2) selection of applicable provisions of the LSC
(3) application of those provisions to a particular situation

A frame-based representation was used for the floorplan and its corresponding objects. The knowledge-based development tool **KEE** runs on special-purpose AI (LISP) development

hardware, and offers a broad range of capabilities, but it can be fairly expensive, often costing tens of thousands of dollars. The tool offers *Frame* representation coupled with *Production rule* representation. The inference engine such as backward chaining, forward chaining, depth-first search, or breadth-first search, could be programmed but is not provided as part of the development tool. Compared with our system, their system is more sophisticated in terms of graphic and computer-aided design and drafting (CADD) capabilities. An actual building floor can be analysed for conformance with the Life Safety Code. In the KEE tool, the basic data element is called a frame . Each frame has a name, a number of slots, and each slot may have a number of facets. Each rule has to be inserted as a procedure in the *EXTERNAL.FORM* slot that contains the rule as the user wrote it, and a *PARSE* method that converts the rule into an internal form consisting of lists of expressions that are values of the *PREMISE* and *ACTION* slots. Dym's system implements the requirements of the Life Safety Code of the National Fire Protection Association in the form of 'if/then' rules. Their system is a comprehensive knowledge-based expert system for automated architectural code checking. The input to their system is a file (knowledge base) that contains information describing a single floor of a building. The output of their system is a floorplan conformance report that contains a list of components of the

floor that do not conform to requirements of the Life Safety Code. Their forward chaining inference mechanism can be combined with the logical inference mechanism. The application of the logical inference mechanism to queries of legal documents can supplement the forward chaining inference mechanism of the existing expert system. The combination of forward chaining and logical inference mechanisms will increase the power of the existing expert system. The user will be able to know all the conditions that have solutions, as well as the solution itself, as we gave examples in chapter 4. Our system enables the user to extract all the relationships between the data items from the knowledge bases. If no solution exists, the system responds with the statement " No more solutions ". For example, for the query concerning the fire regulation provisions, the corresponding values of the building area, the number of storeys, the status whether the building is sprinklered or not, and the number of streets the building faces, are displayed in the output, as well as the solution itself, i.e, the fire regulation provision in the code. The logical inferencing mechanism is useful in cases where the user has a partial knowledge of the physical constraints. We can consider the following strategy to implement the combination of the logical query system with Dym's expert system. An additional knowledge base should be implemented in a logic programming language similar to the one

implemented in this study. For example, a new frame " FIRE_REGULATION.RULE " defines the rules for fire regulation, and will have a RULE slot which contains a rule based on the logical inference mechanism. The values of the slots for all instantiated variables are inherited from the frame " Query " which contains the slots arbitrarily chosen by the user. The new combined system enables the user to have access to the feature of the logical inference query as well as the features of the existing expert system. The additional feature of logical inferencing gives the user all the conditions of the physical constraints or data items, as well as the solution itself, even though the user might have a partial knowledge of the physical constraints. In other words, the logical inference mechanism shows which conditions are possible, and which conditions are impossible in the code. For example, because of the graphic capability of Dym's expert system, the user can point to a structural component such as " WALL X " on the floorplan displayed on the monitor, where X is the wall label. He can then query the system by defining his constraint such as the fire resistance rating of the wall. The system uses the logical inference mechanism and responds with all the possible conditions and solutions for that particular constraint. For example, one of the solutions is the maximum building area allowed in the code, and the conditions are the corresponding fire resistance ratings of other structural

components such as the roof assembly and the floor assembly.

Rasdorf and Parks [18] did research at North Carolina State University on natural language prototypes for analyzing design standards. They transformed the standards written in natural language into computer data structures. Their system consists of three components :

(1) Parser - The parser converts a sentence from a provision of the Code into a case grammar representation of verbs and prepositional phrases. For example, the sentence " Every room or space intended for human occupancy shall be provided with natural or artificial light.", is transformed into the following frames :

prepositional-phrase frame

```
prep          noprep  ;  subject, since no preposition
adjmod        intended ; modifier is adjectives or adverb
dataobject     space  ;     object is a known data item
conj           or;          conjunction
dataobject     room
det            every ;      determiner
add            g101 ;  address indicates the frame's
                       position within the sentence
```

verb-complex frame

```
main_verb     provided ;  always the last verb
modal         shall
past-p        be    ;     past participle
addr          g103
```

(2) Recase - The Reverse Case grammar analyzer transforms the case grammar representation of a sentence into a triplets consisting of a subject data item, an object data

item, and the relacionship between them, in a semantic network.

(3) Query - The query subsystem answers natural-language queries with the number of the corresponding provisions of the Code.

They proposed to expand their system to include an expert system rule generator to transform the recase output into production system rules. Their system is based on natural language and is not suitable for numeric variables and computations. Provisions which are written in pure natural language devoid of numerical values, can be processed using their system. However, their system cannot obviously handle numeric variables, and therefore, it cannot process the fire regulation provisions of the building code. The code focuses on the building area, number of storeys, and the number of streets the building faces. For example, provision 3.2.2.20 of the code states that the building area has a building area not more than

(i) 400 m2 if facing 1 street,

(ii) 500 m2 if facing 2 streets, or

(iii) 600 m2 if facing 3 streets.

Such kinds of logical statements cannot be parsed by Rasdorf and Park's system. However, their user-interface is friendlier than that of our system. The user can question

their system using English sentences, and the system converses with the user in a natural language.

5.4 Prolog-based expert systems

Rosenman and Gero's system was the first Prolog-based system for conformance checking of a building code. They implemented their system in Prolog-1 in a MS-DOS environment. They used a production system, and their system indexes all the objects in the knowledge base, and therefore involves a lot of processing on the part of the knowledge engineer. An index is created consisting of each separate object and the rules in which it appears, both in the antecedent and consequent parts. The index is of the form :

$object_i$ *([ALIST],[CLIST])*

where *ALIST* is the list of rules where $object_i$ appears in the antecedent part and *CLIST* is the list of rules where $object_i$ appears in the consequent part.

The consequent part of their production rule allows for a conjunction of predicates. This has the merit of treating directly the connective **and** in the head of the rule clauses. Rosenman and Gero's system is not menu-driven. The user has to question the system to find out the available knowledge present in the system. Our system is more user-friendly, in that a menu is presented to the user who is able to choose the sections in the code he is interested in. This was

accomplished using a blackboard approach. The knowledge bases are partitioned in our system, and each knowledge base corresponds to a set of clauses in the code. Besides, our system uses not only forward chaining inference engine but logical inference. The logical inference allows the user to extract the possible relationships among the data items, based on the constraints defined by the user. This type of constraint-based query was not implemented in Rosenman and Gero's system. Their system allows for simple queries like *goal(g1('fire resistance rating required'))* or *find 'fire resistance rating required'*. They have pointed out that the environment on which the prototype was developed is not capable of supporting any large scale expert system, in terms of facilities and capacity.

5.5 Semantic Network

A semantic network is a net or graph of nodes joined by links. The nodes in a semantic network represent concepts or meanings and the links represent relations. Knowledge representation based on semantic network has been reviewed and described in details by Harmon and King [14]. The semantic network is very flexible to accomodate new nodes and links which can be defined as needed. Besides, it possesses the important feature of inheritance. Inheritance refers to the ability of one node to " inherit " characteristics of other nodes that are related to it.

G. Redey et al. [19] have built an expert system for the Hungarian Building Regulations. Their three-year project started in 1986, and the implementation language was Prolog for the backward-chaining inference engine, in a semantic network. Similar to Rasdorf and Park's system, they developed a natural language prototype to process the building code. The user communicates in the same language with the system which is to infer all the conclusions corresponding to a given question. A sample query in their system is as follows :

Query : ? {MINIMAL_WIDTH ? piece_of_ground} ?

User : 20

Their system uses a pure semantic network, and therefore lacks the rule format that is applicable for code provisions. They implemented the forward-chaining part in Pascal, which has the merit of being faster than a Prolog-based inference engine. Our system uses Prolog for both forward-chaining and logical inferences, in order to be consistent and coherent.

CHAPTER 6

CONCLUSIONS

6.1 Concluding remarks

We conclude that the use of a logic programming language such as Prolog, is one of the best methods of querying legal documents. We have shown that the combination of logical inferencing and forward chaining mechanisms produces a powerful tool for processing legal documents or codes. The use of expert systems based on forward chaining mechanism can be enhanced by the logical inferencing mechanism when the user has a partial knowledge of the constraints, variables or data items in the legal codes. A prototype expert system shell has been developed in Quintus Prolog, based on a blackboard method of problem-solving. It describes the XSHELL program which can be used as a decision aid in building design.

(a) XSHELL was developed on the Apollo workstation utilising Quintus Prolog in the Unix environment. Future work should attempt to convert the program into MS-DOS environment.

(b) New procedures for new clauses and regulations in the National Building Code, can be easily incorporated in the XSHELL program.

(c) XSHELL could be used as the ultimate design program incorporating CAD (computer-aided drawings), design and code compliance checking for the design of buildings.

The menu-driven system implements not only forward-chaining but logical inferences. The prototype system implements the major regulations applying to Part 3 of the National Building Code. The size and occupancy requirements for fire safety, the safety requirements for fire safety, the safety requirements within floor areas, requirements for exits, and health requirements as applied to plumbing facilities are implemented. These regulations are given as a list of items on a main menu, and it is up to the user to choose the item he is interested in. The item " Term Definition " has been incorporated in the menu to help the user who is not familiar with the technical terms associated with the building code. The knowledge bases are partitioned, so that each knowledge base corresponds to a set of provisions in the code. The system can easily be expanded to accomodate new building regulations, and easily updated to satisfy any future changes in the building regulations.

The Prolog-based representation of design codes enables the user to extract the relationships between data items or variables, subject to user-defined constraints and preferences. Examples are given to display the expressive power of logic to find all possible relationsnips between

the number of storeys and building area for fire regulation clauses in the code. By using the information about the user constraints and preferences, the system proposes the code clauses that conform to all constraints. The proposed system has been compared with traditional methods of knowledge representation for design codes. The limitations of the traditional methods of knowledge representation, such as decision tables and production rules, as applied to building codes, have been shown with respect to Horn clause logic representation.

6.2 Limitations of the system

First, the explanation facility in the user-interface in our system is rather primitive, as it explains the rules in terms of the procedures *parm* and *parmset* used in the expert system shell. The explanation facility can be improved with the user deciding the detailed nature of the explanation. In this sense, the user-interface of our system is not very user-friendly. Besides, the user does not have the capability of interrupting the present system to request various types of information. In the present system, the user is compelled to go back to the main menu, when the system has responded to his query.

Second, the system lacks graphic and computer-aided design and drafting (CADD) capabilities. The interface with a graphic software package such as ' AUTOCAD ', is highly

desirable as the various building structural components such as walls, floors, can easily be accessed and defined. Besides, the present system does not possess sophisticated numerical capabilities as it is Prolog-based. For design codes involving numerical computations, a procedure for interfacing with another high-level language such as Fortran, should be implemented.

6.3 Future developments

The requirements obtained from the building regulations expert system can be used as constraints or parameters for the selection or design of building elements. Commercial software , e.g SFRAME, for structural analysis of building frames or components, can be incorporated in the blackboard architecture as stand-alone knowledge sources. The design of the building components, i.e, sizing and proportioning of the members could be performed by selecting a member of a database of Canadian standard sections. The section is then checked for compliance with provisions of relevant existing Canadian design standards, e.g, CSA Standard CAN3-A23.3-M84 " Design of Concrete Structures for Buildings".
This approach has been implemented by K. Gowri et al. [20] for a knowledge-based system for the analysis and design of aluminum structural components according to the Canadian standard CAN3-S157-M83. A hybrid knowledge representation of frames and rules was implemented using the commercial tool

GOLDWORKS and user-defined GCLISP functions. The knowledge base consists of frames, instances, demon functions and rules for representing the design context and code specifications. For example, the material properties of the alloys and geometrical details are encoded in frames. Frames may be implemented in Prolog as follows :

```
frame(name,[
        slotname1 : [ facet1(value1), facet2(value2),... ],
        slotname2 : [ facet1(value1), facet2(value2),... ],
                    ..........]).
```

The building frame can have the following specification :

```
frame(building,[
        use : [ 'Group A','Group B','Group C','Group D',
                'Group E','Group F'],
        parts_of : [ foundation,column,wall,floor,roof],
        storeys : [ integer]]).
```

The child frame 'wall' can be defined as follows :

```
frame(wall,[
        location : [ _ ],
```

material : [concrete,masonry,..],

inheritance : [building]]).

The class inheritance property of a frame-based representation makes efficient use of storage and minimizes data redundancy.

Frames provide an efficient scheme for engineering design problems which are hierarchical by nature.

We can safely predict that a hybrid system of frames and rules will in the future , be the scheme to automate the building design process from conceptual design through construction planning.

We need to make a more user-friendly interface based on graphics as explained in the previous section. An interface with a graphics commercial software such as " AUTOCAD " is highly desirable.

REFERENCES

1. National Building Code of Canada, 1990
   National Research Council of Canada, Ottawa,Ontaric
2. Quintus Prolog Release 2.4 (UNIX)
   Quintus Computer Systems
   Mountain View, California

3. M.A. Rosenman and J.S.Gero : Design codes as expert systems, Computer-aided design vol 7 number 2 november 1985,Butterworth & Co (Publishers) Ltd

4. M.A.Covington, D.Nute, A.Vellino : Prolog Programming in Depth, Scott,Foresman and Company,1988

5. P.Fazio,C.Bedard,K.Gowri : Knowledge-Based System Development Tools for Processing Design Specifications, Microcomputers in Civil Engineering,3(4),333-344 (1988)

6. Maher,M.L HI-RISE : a knowledge-based expert system for the preliminary structural design of high rise buildings, Ph D thesis,Department of Civil Engineering, Carnegie-Mellon University (1984)

7. Guru

Micro Data Base Systems Inc

8. Personal Consultant Plus

Texas Instruments Inc

9. Goldworks

Gold Hill Computers

10. Reddy,D.R, L.D.Erman,R.B.Neeley : A Model and a System for Machine Recognition of Speech, IEEE Transactions on Audio and Electroacoustics, Vol. AU-21,1973, pp. 229-238

11. Nii,H.P. and N.Aiello : AGE : A knowledge-Based Program for Building Knowledge-Based Programs, Proceedings of the 6th International Joint Conference on Artificial Intelligence, William Kaufmann Inc, Los Altos, Calif., 1979,pp. 645-655

12. Nii,H.P. : Blackboard Systems : The Blackboard Model of Problem Solving and the Evolution of Blackboard Architectures,The AI Magazine, Vol.7,No.2,summer 1986, pp 38 - 53

13. Nii,H.P. : Blackboard Systems : Blackboard Application Systems,Blackboard Systems from a Knowledge Engineering

Perspective,The AI Magazine,Vol.7,No. 3, August 1986, pp 82 - 106

14. P. Harmon and D. King : Expert Systems - Artificial Intelligence in Business, 1985, John Wiley & Sons, chapter 4

15. W.J.Rasdorf,T.E.Wang : Generic Design Standards Processing in an expert system environment,Journal of Computing in Civil Engineering,Vol 2,N0.1,January 1988, pp 68-87

16. Forgy, C.L : OPS5 User's Manual, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Pa., 1981

17. Dym,C.L,R.P.Henchey,E.A.Delis,S.Gonick : A knowledge-Based System for Automated Architectural Code Checking,Computer-Aided Design Journal,Vol 20,No.3, April 1988,pp 137 - 145

18. W.J.Rasdorf,L.M.Parks : Natural Language Prototypes for Analyzing Design Standards,Artificial Intelligence in Engineering:Tools and Techniques,Computational Mechanics Publications,Southampton,UK,1987,pp 147 - 160

19. G.Redey,E.G.Nagy,P.Z.Georgieva:An expert system approach to the Hungarian Building Regulations, Network Information Processing Systems,Proceedings of the IFIP TC6/Tc8 Open Symposium,Sofia,Bulgaria,May 1988, pp 321 - 328

20. K. Gowri,C.Marsh,C.Bedard,P.Fazio : Knowledge-based assistant for aluminum component design, Computers & Structures , Vol. 38,No. 1,pp. 9-20,1991.

8.1 APPENDIXES

PROGRAM LISTINGS

```
/* XSHELL.PL */
/*
 * An expert system consultation driver to be used
 * with separately written knowledge bases.
 *
 * Procedures in the file include xshell, xshell_aux,
 * finish_xshell, prop, parm, parmset, parmrange,
 * explain, member, and wait.
 *
 * Requires various procedures defined in the files
 * READSTR.PL,READNUM.PL,WRITELN.PL, and YES.PL
 *
 */


:- ensure_loaded('readint.pl').
:- ensure_loaded('writeln.pl').
:- ensure_loaded('yes.pl').
:- dynamic known/2,prop/1,parm/3,parmrange/3,parmset/3.
:- dynamic xkb_question/2,xkb_identify/1.
:- dynamic xkb_define/1,xkb_fireregulation/1,xkb_safety/1.
:- multifile xkb_question/2,xkb_identify/1.
:- multifile xkb_define/1,xkb_fireregulation/1,xkb_safety/1.
:-multifil exshell/0,xshell1/0,xshell2/0,xshell3/0,xshell4/0,
   xshell5/0.
:- multifile xshell6/0,xkb_plumbing/1.
:- retractall(known(_,_)).
:- retractall(prop(_)).
:- retractall(parm(_,_,_)).
:- retractall(parmrange(_,_,_)).
:- retractall(parmset(_,_,_)).



/*
 * xshell
 * As a query, this predicate begins a consultation. It is
 * the main program or procedure for the expert system
 * consultation driver. It always succeeds.
 */


xshell :-
          xkb_intro(Statement),
          writeln(Statement), nl,
          xkb_identify(ID),
          asserta(known(identification,ID)),
          xkb_report(Phrase),
          write(Phrase),
          writeln(ID),nl,nl,nl,
          explain,
```

```
          xkb_unique(yes),
          !,
          xshell_aux.


xshell1 :-
          xkb_fireregulation(FIRE),
          assert(known(fireregulation,FIRE)),
          xkb_report1(Phrase1),
          write(Phrase1),nl,
          writeln(FIRE),nl,
          explain1,
          xkb_unique(yes),
          !,
          xshell1_aux.


xshell2 :-
          xkb_firealarm(Alarm),
          assert(known(firealarm,Alarm)),
          writeln(Alarm),nl,
          !,
          finish2_xshell.



xshell3 :-
          xkb_define(Definition),
          assert(known(termdefinition,Definition)),
          writeln(Definition),nl,
          !,
          finish2_xshell.

xshell4 :-
          xkb_safety(Safe),
          assert(known(safety_floor_areas,Safe)),
          writeln(Safe),nl,
          explain1,
          xkb_unique(yes),
          !,
          xshell2_aux.

xshell5 :-
          xkb_exit(Exit),
          assert(known(exit_floor_areas,Exit)),
          writeln(Exit),nl,
          explain1,
          xkb_unique(yes),
          !,
          xshell2_aux.
```

```
xshell6 :-
          xkb_plumbing(Plumbing),
          assert(known(plumbing_facilities,Plumbing)),
          writeln(Plumbing),nl,
          explain1,
          xkb_unique(yes),
          !,
          xshell2_aux.


xshell :- xshell_aux.

xshell1 :- xshell1_aux.
xshell4 :- xshell2_aux.
xshell5 :- xshell3_aux.
xshell6 :- xshell4_aux.

/*
 * xshell_aux
 * Prevents an abrupt end to a consultation that ends
 * without an identification, or a consultation where
 * multiple identifications are allowed.
 */


xshell_aux :- \+ known(identification,_),
               writeln(' I cannot reach a conclusion.'),
              !,
              finish_xshell.

xshell_aux :- xkb_unique(no),
              known(identification,_),
              writeln('I    cannot    reach    any    further
conclusion.'),
               !,
              finish_xshell.


xshell1_aux :-
              \+ known(fireregulation,_),
              writeln(' I cannot reach a conclusion.'),
              !,
              finish1_xshell.

xshell1_aux :- xkb_unique(no),
              known(fireregulation,_),
              writeln('I    cannot    reach    any    further
conclusion.'),
              !,
              finish1_xshell.
```

```
xshell2_aux :- xkb_unique(no),
               known(safety_floor_areas,_),
               writeln('I    cannot    reach    any    further
conclusion.'),
               !,
               finish1_xshell.


xshell3_aux :- xkb_unique(no),
               known(exit_floor_areas,_),
               writeln('I    cannot    reach    any    further
conclusion.'),
               !,
               finish1_xshell.


xshell4_aux :- xkb_unique(no),
               known(plumbing_facilities,_),
               writeln('I    cannot    reach    any    further
conclusion.'),
               !,
               finish1_xshell.


xshell_aux :- finish_xshell.
xshell1_aux :- finish1_xshell.
xshell2_aux :- finish1_xshell.

/*
 * finish_xshell
 * Erases the working database and asks if the user wants
 * to conduct another consultation.
 */

finish_xshell :-
     retractall(known(_,_)),
     writeln('Do you want to conduct another consultation?'),
    yes('>'), nl, nl,
    !,
    xshell.

finish_xshell.


finish1_xshell :-
     retractall(known(_,_)),
     writeln('Do you want to conduct another consultation?'),
    yes('>'), nl, nl,
    !,
    xshell1.
```

```
finish1_xshell.
finish2_xshell :-
          retractall(known(_,_)).

/*
 * prop(Property)
 * Succeeds if it is remembered from an earlier call that
 * the subject has the property. Otherwise the user is
 * asked if the subject has the property and the user's
 * answer is remembered. In this case, the procedure call
 * succeeds only if the user answers 'yes'.
 */

prop(Property) :- known(Property,Value),
                  !,
                  Value==y.

prop(Property) :- xkb_question(Property,Question),
                  writeln(Question),
                  yes('>'),nl,nl,
                  assert(known(Property,y)),
                  !.

prop(Property) :- assert(known(Property,n)),
                  nl,nl,
                 !,
                  fail.



/*
 * parm(Parameter,Type,Value)
 *  Type determines whether Value is to be a character, an
 *  atom, or a number. Value becomes the remembered value
 *  for the parameter if there is one. Otherwise the user is
 *  asked for a value and that value is remembered. When
 *  used as a test condition, Value is instantiated before
 *  the procedure is called and parm(Parameter,Type,Value)
 *  only succeeds if the remembered value, or alternatively
 *  the value reported by the user, matches Value. */

parm(Parameter,_,Value) :- known(Parameter,StoredValue),
                           !,
                           Value=StoredValue.

parm(Parameter,c,Value) :- xkb_question(Parameter,Question),
                           writeln(Question),
                           write('>'),
                           get(Char),nl,nl,
                           name(Response,[Char]),
                          assert(known(Parameter,Response)),
                           !,
```

```
                              Value=Response.


parm(Parameter,a,Value)  :- xkb_question(Parameter,Question),
                            writeln(Question),
                            readatom(Response),nl,nl,
                            assert(known(Parameter,Response)),
                            !,
                            Value=Response.

parm(Parameter,n,Value)  :- xkb_question(Parameter,Question),
                            writeln(Question),
                            readinteger(Response),nl,nl,
                            assert(known(Parameter,Response)),
                            !,
                            Value=Response.

/*
 * parmset(Parameter,Type,Set)
 *  Type indicates whether Parameter takes a character,
 *  an atom, or a number as value, and Set is a list of
 *  possible values for Parameter. A call to the procedure
 *  succeeds if a value for Parameter is established that is
 *  a member of Set.
 */

parmset(Parameter,Type,Set)  :- parm(Parameter,Type,Value),
                                member(Value,Set).

/*
 * parmrange(Parameter,Minimum,Maximum)
 *   Parameter should be a parameter that takes numbers as
 *   values, and Minimum and Maximum should be numbers. A
 *   call to the procedure succeeds if a value for Parameter
 *   is established that is in the closed interval
 *   [Minimum,Maximum].
 */

parmrange(Parameter,Minimum,Maximum)  :-
                    parm(Parameter,n,Value),
                    Minimum=<Value,
                    Maximum>=Value.


/* explain
 *  Explains how the expert system arrived at a conclusion
 *  by first finding an identification rule for the conclusion
in
 *   the knowledge base whose condition succeeds, and then
showing
 *  it to the user. If xkb_explain(no) is in the knowledge
```

```
 *  base, explain merely waits for a keystroke to give the
 *  user time to read the conclusion.
 */

explain :- xkb_explain(no),wait,!.

explain :- writeln(
           ['Do you want to see the rule that was used',
           'to reach the conclusion?']),
            \+ yes('>'),nl,!.

explain :- known(identification,ID),
           clause(xkb_identify(ID),Condition),
           Condition,nl,nl,
           write('Rule: '),
           xkb_report(Phrase),
           write(Phrase),
           writeln(ID),
           writeln('   if '),
           write(Condition),nl,nl,!.


explain1 :- xkb_explain(no),wait,!.

explain1 :- writeln(
           ['Do you want to see the rule that was used',
           'to reach the conclusion?']),
            \+ yes('>'),nl,!.



explain1 :- known(fireregulation,FIRE),
           clause(xkb_fireregulation(FIRE),Condition),
           Condition,nl,nl,
           write('Rule: '),
           xkb_report1(Phrase1),
           write(Phrase1),
           writeln(FIRE),
           writeln('     if '),
           write(Condition),nl,nl,!.


/*
 * wait
 *   Prints prompt and waits for keystroke.
 */

wait :- write('Press Return when ready to continue. '),
        get(_),nl,nl.

member(X,[X|_]).
```

```
member(X,[_|Y]) :- member(X,Y).

end_of_file.
```

```
/*
 * BUILDING.PL  */

/*
 * Contains an XSHELL knowledge base.
 * Requires all procedures in XSHELL.PL.
 */



:- ensure_loaded('xshell.pl').
:- ensure_loaded('firealarm.pl').
:- ensure_loaded('safety.pl').
:- ensure_loaded('exit.pl').
:- ensure_loaded('plumbing.pl').
:- ensure_loaded('definition.pl').
:- ensure_loaded('fire1_for.pl').
:- ensure_loaded('fire3_for.pl').
:- ensure_loaded('fire5_for.pl').
:- ensure_loaded('fire6_for.pl').



/*
 * Any clauses for the predicates xkb_intro, xkb_report,
 * xkb_unique, xkb_explain, xkb_identify, and
 * xkb_question should be removed from the knowledge base.
 */
:- dynamic xkb_intro/1,xkb_report/1,xkb_unique/1.
:- dynamic xkb_explain/1.
:- dynamic xkb_report1/1.
:- dynamic rule/5.
:- multifile rule/5.

:- retractall(xkb_intro(_)).
:- retractall(xkb_report(_)).
:- retractall(xkb_unique(_)).
:- retractall(xkb_explain(_)).
:- retractall(xkb_report1(_)).
:- retractall(rule(_,_,_,_,_)).
:- ensure_loaded('fire_regulation.pl').


xkb_intro(['',

   ' BUILDING : An Expert System for building regulations ',

   '              in the National Building Code of Canada',


   '',
   ' This program will help you identify the building
```

```
      regulations in the National Building Code of Canada
      (1990).',
    ' First , the classification of the building is
      identified based on its use and occupancy.',
  '',
    ' To use the program, simply describe the building by',
    'answering the following questions.']).


xkb_report(' Building identification : ').
xkb_report1(' Fire regulation :  ').
xkb_unique(yes).
xkb_explain(yes).


/*
 *  xkb_identify(Building)
 *  Each clause for this predicate provides a rule
 *  to be used with the utility predicates in
 *  the XSHELL.PL file to identify the building.
 */


/*  Rule 1   */
xkb_identify(' Group A Division 1 ') :-
                          prop(assembly_occupancy),
                          prop(performing_arts).


/*  Rule 2   */

xkb_identify(' Group A Division 3 ') :-
                           prop(assembly_occupancy),
                           prop(arena_type).


/*  Rule 3  */

xkb_identify(' Group A Division 4 ') :-
                           prop(assembly_occupancy),
                           prop(open_air).

/*  Rule 4  */

xkb_identify(' Group A Division 2 ') :-
                           prop(assembly_occupancy),
                       \+ prop(performing_arts),
                       \+ prop(arena_type),
                       \+ prop(open_air).
```

```
/* Rule 5 */

xkb_identify(' Group B Division 1 ') :-
                              prop(institutional_occupancy),
                              prop(penal_detention).

xkb_identify(' Group B Division 2 ') :-
                              prop(institutional_occupancy),
                            \+ prop(penal_detention).


/* Rule 6 */

xkb_identify(' Group C ') :-
                              prop(residential_occupancy).


/* Rule 7  */

xkb_identify(' Group D ') :-
                            prop(business_occupancy).

/* Rule 8 */

xkb_identify(' Group E ') :-
                            prop(mercantile_occupancy).

/* Rule 9  */

xkb_identify(' Group F Division 1 ') :-
                           prop(industrial_occupancy),
                           prop(high_combustible_content).

/* Rule 10  */
xkb_identify(' Group F Division 2 ') :-
                           prop(industrial_occupancy),
                           prop(medium_combustible_content).

/* Rule 11  */

xkb_identify(' Group F Division 3 ') :-
                        prop(industrial_occupancy),
                    \+ prop(high_combustible_content),
                    \+ prop(medium_combustible_content).



/* FIRE REGULATIONS */


/*
```

```
*   X = number of storeys used for logical inference rules
*   Y = building area
*   Z = number of streets the building faces
*   A = rule number
*   S = status whether the building is sprinklered or not
*   S = 1 (unsprinklered)
*   S = 2 (sprinklered)
*/


          rule(12,S,X,Y,Z) :-
                      xkb_identify(' Group A Division 1 '),
                         X = 1,
                         prop(auditorium_floor_height),
                         parm(occupant_load_floor,c,a).


         rule(13,S,X,Y,Z) :-
                     xkb_identify(' Group A Division 1 '),
                       X = 1,
                       parm(occupant_load_floor,c,b),
                       parm(building_area,c,1).



         rule(14,S,X,Y,Z) :-
                      xkb_identify(' Group A Division 1 '),
                          X = 'unlimited',
                          Y = 'unlimited',
                          \+ known(fireregulation).


        rule(15,S,X,Y,Z) :-
                    xkb_identify(' Group A Division 2 '),
                    X = 1,

         ((S = 1,
              (((Z = 1,var(Y),Y = 400)
               | (Z = 1,nonvar(Y), Y =< 400))
            |  ((Z = 2,var(Y),Y = 500)
               |(Z = 2,nonvar(Y),Y =< 500))
            |  ((Z = 3,var(Y),Y = 600)
               |(Z = 3,nonvar(Y),Y =< 600 ))))
        | (S = 2,
              (((Z = 1,var(Y),Y = 800)
               | (Z = 1,nonvar(Y),Y =< 800))
            |  ((Z = 2,var(Y),Y = 1000)
               |(Z = 2,nonvar(Y),Y =< 1000))
            |  ((Z = 3,var(Y),Y = 1200)
               |(Z = 3,nonvar(Y),Y =< 1200))))).
```

```
rule(16,S,X,Y,Z) :-
            xkb_identify(' Group A Division 2 '),
            (X = 1 | X = 2),
            S = 2,
          (((Z = 1,var(Y),Y = 400)
            | (Z = 1,nonvar(Y),Y =< 400))
        |  ((Z = 2,var(Y),Y = 500)
            | (Z = 2,nonvar(Y),Y =< 500))
        |  ((Z = 3,var(Y),Y = 600)
            | (Z = 3,nonvar(Y),Y =< 600))).


          rule(17,S,X,Y,Z) :-
                xkb_identify(' Group A Division 2 '),
                   (( X = 1,
((S = 1,
        (((Z = 1,var(Y),Y = 1600)
          | (Z = 1,nonvar(Y),Y =< 1600,Y > 400))
        | ((Z = 2,var(Y),Y = 2000)
          | (Z = 2,nonvar(Y),Y =< 2000,Y > 500))
        | ((Z = 3,var(Y),Y = 2400)
          | (Z = 3,nonvar(Y),Y =< 2400,Y > 600))))
  | (S = 2,
        (((Z = 1,var(Y),Y = 3200)
          | (Z = 1,nonvar(Y),Y =< 3200,Y > 800))
        |  ((Z = 2,var(Y),Y = 4000)
            | (Z = 2,nonvar(Y),Y =< 4000,Y > 1000))
        |  ((Z = 3,var(Y),Y = 4800)
            | (Z = 3,nonvar(Y),Y =< 4800,Y > 1200))))))

        |           (  X = 2,
((S = 1,
          (((Z = 1,var(Y),Y = 800)
            | (Z = 1,nonvar(Y),Y =< 800))
        |  ((Z = 2,var(Y),Y = 1000)
            | (Z = 2,nonvar(Y),Y =< 1000))
        |  ((Z = 3,var(Y),Y = 1200)
            | (Z = 3,nonvar(Y),Y =< 1200))))
  | (S = 2,
          (((Z = 1,var(Y),Y = 1600)
            | (Z = 1,nonvar(Y),Y =< 1600,Y > 400))
        |  ((Z = 2,var(Y),Y = 2000)
            | (Z = 2,nonvar(Y),Y =< 2000,Y > 500))
        |  ((Z = 3,var(Y),Y = 2400)
            | (Z = 3,nonvar(Y),Y =<  2400,Y >
                        600))))))).


rule(18,S,X,Y,Z) :-
                xkb_identify(' Group A Division 2 '),
```

```
                X = 1,

((S = 1,
          (((Z = 1,var(Y),Y = 800)
           | (Z = 1,nonvar(Y), Y =< 800,Y > 400))
        |  ((Z = 2,var(Y),Y =1000)
           |(Z = 2,nonvar(Y),Y =< 1000,Y > 500))
        |  ((Z = 3,var(Y),Y = 1200)
           |(Z = 3,nonvar(Y),Y =< 1200,Y > 600 ))))
   | (S = 2,
          (((Z = 1,var(Y),Y = 1600)
           | (Z = 1,nonvar(Y),Y =< 1600,Y > 800))
        |  ((Z = 2,var(Y),Y = 2000)
           |(Z = 2,nonvar(Y),Y =< 2000,Y > 1000))
        |  ((Z = 3,var(Y),Y = 2400)
           |(Z = 3,nonvar(Y),Y =< 2400,Y > 1200))))).


          rule(19,S,X,Y,Z) :-
                xkb_identify(' Group A Division 2 '),

                   (( X = 1,

((S = 1,
        (((Z = 1,var(Y),Y = 1600)
          | (Z = 1,nonvar(Y),Y =< 1600,Y > 800))
        | ((Z = 2,var(Y),Y = 2000)
          | (Z = 2,nonvar(Y),Y =< 2000,Y > 1000))
        | ((Z = 3,var(Y),Y = 2400)
          | (Z = 3,nonvar(Y),Y =< 2400,Y > 1200))))
   | (S = 2,
         (((Z = 1,var(Y),Y = 3200)
          | (Z = 1,nonvar(Y),Y =< 3200,Y > 1600))
        |  ((Z = 2,var(Y),Y = 4000)
          | (Z = 2,nonvar(Y),Y =< 4000,Y > 2000))
        |  ((Z = 3,var(Y),Y = 4800)
          | (Z = 3,nonvar(Y),Y =< 4800,Y > 2400))))))

        |            (  X = 2,
((S = 1,
         (((Z = 1,var(Y),Y = 800)
          | (Z = 1,nonvar(Y),Y =< 800))
        |  ((Z = 2,var(Y),Y = 1000)
          | (Z = 2,nonvar(Y),Y =< 1000))
        |  ((Z = 3,var(Y),Y = 1200)
          | (Z = 3,nonvar(Y),Y =< 1200))))
   | (S = 2,
         (((Z = 1,var(Y),Y = 1600)
          | (Z = 1,nonvar(Y),Y =< 1600,Y > 400))
        |  ((Z = 2,var(Y),Y = 2000)
```

```
                | (Z = 2,nonvar(Y),Y =< 2000,Y > 500))
            |  ((Z = 3,var(Y),Y = 2400)
                | (Z = 3,nonvar(Y),Y =< 2400,Y >
                            600))))))).


          rule(20,S,X,Y,Z) :-
               xkb_identify(' Group A Division 2 '),

                    (( X = 1,

    ((S = 1,
           (((Z = 1,var(Y),Y = 'unlimited')
             | (Z = 1,nonvar(Y),Y > 1600))
           | ((Z = 2,var(Y),Y = 'unlimited')
             | (Z = 2,nonvar(Y),Y > 2000))
           | ((Z = 3,var(Y),Y = 'unlimited')
             | (Z = 3,nonvar(Y),Y > 2400))))
      | (S = 2,
           (((Z = 1,var(Y),Y = 'unlimited')
             | (Z = 1,nonvar(Y),Y > 3200))
           |  ((Z = 2,var(Y),Y = 'unlimited')
             | (Z = 2,nonvar(Y),Y > 4000))
           |  ((Z = 3,var(Y),Y = 'unlimited')
             | (Z = 3,nonvar(Y),Y > 4800))))))

           |          (  X = 2,
    ((S = 1,
           (((Z = 1,var(Y),Y = 'unlimited')
             | (Z = 1,nonvar(Y),Y > 800))
           |  ((Z = 2,var(Y),Y = 'unlimited')
             | (Z = 2,nonvar(Y),Y > 1000))
           |  ((Z = 3,var(Y),Y = 'unlimited')
             | (Z = 3,nonvar(Y),Y > 1200))))
      | (S = 2,
           (((Z = 1,var(Y),Y = 'unlimited')
             | (Z = 1,nonvar(Y),Y > 1600))
           |  ((Z = 2,var(Y),Y = 'unlimited')
             | (Z = 2,nonvar(Y),Y > 2000))
           |  ((Z = 3,var(Y),Y = 'unlimited')
             | (Z = 3,nonvar(Y),Y > 600))))))))
                | X = 3 | X = 4 | X = 5.



rule(21,S,X,Y,Z) :-
                  xkb_identify(' Group A Division 2 '),
                  X = 1,
     ((S = 1,
           ((var(Y),Y = 3200)
```

```
          | (nonvar(Y), Y =< 3200)))
    | (S = 2,
         ((var(Y),Y = 6400)
          | (nonvar(Y),Y =< 6400)))).


    rule(22,S,X,Y,Z) :-
            xkb_identify(' Group A Division 2 '),
            ((var(X),X = 'greater than 5')
             | (nonvar(X),X >= 6)).


          rule(23,S,X,Y,Z) :-
              xkb_identify(' Group A Division 3 '),
                   X = 1,
((S = 1,
        (((Z = 1,var(Y),Y = 1000)
         | (Z = 1,nonvar(Y), Y =< 1000))
       | ((Z = 2,var(Y),Y = 1250)
          |(Z = 2,nonvar(Y),Y =< 1250))
       | ((Z = 3,var(Y),Y = 1500)
          |(Z = 3,nonvar(Y),Y =< 1500 ))))
  | (S = 2,
        (((Z = 1,var(Y),Y = 2000)
         | (Z = 1,nonvar(Y),Y =< 2000))
       | ((Z = 2,var(Y),Y = 2500)
          |(Z = 2,nonvar(Y),Y =< 2500))
       | ((Z = 3,var(Y),Y = 3000)
          |(Z = 3,nonvar(Y),Y =< 3000))))).


    rule(24,S,X,Y,Z) :-
            xkb_identify(' Group A Division 3 '),

                   X = 1,
((S = 1,
       (((Z = 1,var(Y),Y = 2400)
         | (Z = 1,nonvar(Y),Y =< 2400,Y > 1000))
      | ((Z = 2,var(Y),Y = 3000)
         | (Z = 2,nonvar(Y),Y =< 3000,Y > 1250))
      | ((Z = 3,var(Y),Y = 3600)
         | (Z = 3,nonvar(Y),Y =< 3600,Y > 1500))))
  | (S = 2,
       (((Z = 1,var(Y),Y = 4800)
         | (Z = 1,nonvar(Y),Y =< 4800,Y > 2000))
       | ((Z = 2,var(Y),Y = 6000)
         | (Z = 2,nonvar(Y),Y =< 6000,Y > 2500))
       | ((Z = 3,var(Y),Y = 7200)
         | (Z = 3,nonvar(Y),Y =< 7200,Y > 3000))))).
```

```
rule(25,S,X,Y,Z) :-
                    xkb_identify(' Group A Division 3 '),
                    X = 1,

     ((S = 1,
               (((Z = 1,var(Y),Y = 1200)
                | (Z = 1,nonvar(Y), Y =< 1200))
             |  ((Z = 2,var(Y),Y =1500)
                |(Z = 2,nonvar(Y),Y =< 1500))
             |  ((Z = 3,var(Y),Y = 1800)
                |(Z = 3,nonvar(Y),Y =< 1800 ))))
        | (S = 2,
               (((Z = 1,var(Y),Y = 2400)
                | (Z = 1,nonvar(Y),Y =< 2400))
             |  ((Z = 2,var(Y),Y = 3000)
                |(Z = 2,nonvar(Y),Y =< 3000))
             |  ((Z = 3,var(Y),Y = 3600)
                |(Z = 3,nonvar(Y),Y =< 3600))))).


rule(26,S,X,Y,Z) :-
                    xkb_identify(' Group A Division 3 '),
                    X = 1,

     ((S = 1,
               (((Z = 1,var(Y),Y = 2400)
                | (Z = 1,nonvar(Y), Y =< 2400,Y > 1200))
             |  ((Z = 2,var(Y),Y =3000)
                |(Z = 2,nonvar(Y),Y =< 3000,Y > 1500))
             |  ((Z = 3,var(Y),Y = 3600)
                |(Z = 3,nonvar(Y),Y =< 3600,Y > 1800 ))))
        | (S = 2,
               (((Z = 1,var(Y),Y = 4800)
                | (Z = 1,nonvar(Y),Y =< 4800,Y > 2400))
             |  ((Z = 2,var(Y),Y = 6000)
                |(Z = 2,nonvar(Y),Y =< 6000,Y > 3000))
             |  ((Z = 3,var(Y),Y = 7200)
                |(Z = 3,nonvar(Y),Y =< 7200,Y > 3600))))).


   rule(27,S,X,Y,Z) :-
                    xkb_identify(' Group A Division 3 '),

                          (( X = 1,
    ((S = 1,
               (((Z = 1,var(Y),Y = 4000)
                | (Z = 1,nonvar(Y),Y =< 4000,Y > 2400))
             |  ((Z = 2,var(Y),Y = 5000)
                | (Z = 2,nonvar(Y),Y =< 5000,Y > 3000))
             |  ((Z = 3,var(Y),Y = 2400)
```

```
          | (Z = 3,nonvar(Y),Y =< 6000,Y > 3600))))
  | (S = 2,
       (((Z = 1,var(Y),Y = 8000)
         | (Z = 1,nonvar(Y),Y =< 8000,Y > 4800))
     |  ((Z = 2,var(Y),Y = 10000)
         | (Z = 2,nonvar(Y),Y =< 10000,Y > 6000))
     |  ((Z = 3,var(Y),Y = 12000)
         | (Z = 3,nonvar(Y),Y =< 12000,Y >
                                  7200))))))

     |           (  X = 2,
((S = 1,
       (((Z = 1,var(Y),Y = 2000)
         | (Z = 1,nonvar(Y),Y =< 2000))
     |  ((Z = 2,var(Y),Y = 2500)
         | (Z = 2,nonvar(Y),Y =< 2500))
     |  ((Z = 3,var(Y),Y = 3000)
         | (Z = 3,nonvar(Y),Y =< 3000))))
  | (S = 2,
       (((Z = 1,var(Y),Y = 4000)
         | (Z = 1,nonvar(Y),Y =< 4000))
     |  ((Z = 2,var(Y),Y = 5000)
         | (Z = 2,nonvar(Y),Y =< 5000))
     |  ((Z = 3,var(Y),Y = 6000)
         | (Z = 3,nonvar(Y),Y =< 6000)))))))).


   rule(28,S,X,Y,Z) :-
          xkb_identify(' Group A Division 3 '),

                ((  X = 1,
((S = 1,
      (((Z = 1,var(Y),Y = 'unlimited')
        | (Z = 1,nonvar(Y),Y > 4000))
     |  ((Z = 2,var(Y),Y = 'unlimited')
         | (Z = 2,nonvar(Y),Y > 5000))
     |  ((Z = 3,var(Y),Y = 'unlimited')
         | (Z = 3,nonvar(Y),Y > 6000))))
  | (S = 2,
       (((Z = 1,var(Y),Y = 'unlimited')
         | (Z = 1,nonvar(Y),Y > 8000))
     |  ((Z = 2,var(Y),Y = 'unlimited')
         | (Z = 2,nonvar(Y),Y > 10000))
     |  ((Z = 3,var(Y),Y = 'unlimited')
         | (Z = 3,nonvar(Y),Y > 12000))))))

     |           (  X = 2,
((S = 1,
       (((Z = 1,var(Y),Y = 'unlimited')
         | (Z = 1,nonvar(Y),Y > 2000))
     |  ((Z = 2,var(Y),Y = 'unlimited')
         | (Z = 2,nonvar(Y),Y > 2500))
```

```
    |  ((Z = 3,var(Y),Y = 'unlimited')
       | (Z = 3,nonvar(Y),Y > 3000))))
| (S = 2,
     (((Z = 1,var(Y),Y = 'unlimited')
       | (Z = 1,nonvar(Y),Y > 4000))
    |  ((Z = 2,var(Y),Y = 'unlimited')
       | (Z = 2,nonvar(Y),Y > 5000))
    |  ((Z = 3,var(Y),Y = 'unlimited')
       | (Z = 3,nonvar(Y),Y > 6000))))))
| (nonvar(X),X > 2)).


  rule(29,S,X,Y,Z) :-
        xkb_identify(' Group A Division 4 ').



rule(30,S,X,Y,Z) :-
        xkb_identify(' Group B Division 1 ').


    rule(31,S,X,Y,Z) :-
          xkb_identify(' Group B Division 2 '),
               X = 1,
          ((( S = 1,var(Y),Y = 250)
            | (S = 1,nonvar(Y),Y =< 250))
       |  ((S = 2,var(Y),Y = 500)
           | (S = 2,nonvar(Y),Y =< 500))).


    rule(32,S,X,Y,Z) :-
          xkb_identify(' Group B Division 2 '),

             ((X = 1,
              (((S = 1,var(Y),Y = 1000)
                | (S = 1,nonvar(Y),Y =< 1000))
             | ((S = 2,var(Y),Y = 2400)
                | (S = 2,nonvar(Y),Y =< 2400))))
             | ( X = 2,
               (((S = 1,var(Y),Y = 500)
                 | (S = 1,nonvar(Y),Y =< 500))
              | ((S = 2,var(Y),Y = 1600)
                 | (S = 2,nonvar(Y),Y =<
                                   1600))))).


    rule(33,S,X,Y,Z) :-
          xkb_identify(' Group B Division 2 '),
               S = 2,
             ((( X = 1,var(Y),Y = 'unlimited')
               | (X = 1,nonvar(Y),Y > 2400))
             | (( X = 2,var(Y),Y = 12000)
```

```
                    | (X = 2,nonvar(Y),Y > 1600,Y =<
                                        12000))
                | (( X = 3,var(Y),Y = 8000)
                    | (X = 3,nonvar(Y),Y =< 8000))).


    rule(34,S,X,Y,Z) :-
          xkb_identify(' Group B Division 2 '),

                ((X = 1,
                  ((S = 1,var(Y),Y = 'unlimited')
                    | (S = 1,nonvar(Y),Y > 1000)))
               | ( X = 2,
                   (((S = 1,var(Y),Y = 'unlimited')
                    | (S = 1,nonvar(Y),Y > 500))
                | ((S = 2,var(Y),Y = 'unlimited')
                    | (S = 2,nonvar(Y),Y > 12000))))
                | (X = 3,
                  ((S = 2,var(Y),Y = 'unlimited')
                    | (S = 2,nonvar(Y),Y > 8000)))
                | (nonvar(X),X > 3)).


    rule(35,S,X,Y,Z) :-
              xkb_identify(' Group C '),
             (( X = 1,

 ((S = 1,
       (((Z = 1,var(Y),Y = 1800)
          | (Z = 1,nonvar(Y),Y =< 1800))
       | ((Z = 2,var(Y),Y = 2250)
          | (Z = 2,nonvar(Y),Y =< 2250))
       | ((Z = 3,var(Y),Y = 2700)
          | (Z = 3,nonvar(Y),Y =< 2700))))
  |  (S = 2,
      (((Z = 1,var(Y),Y = 3600)
         | (Z = 1,nonvar(Y),Y =< 3600))
      |  ((Z = 2,var(Y),Y = 4500)
          | (Z = 2,nonvar(Y),Y =< 4500))
      |  ((Z = 3,var(Y),Y = 5400)
          | (Z = 3,nonvar(Y),Y =< 5400))))))

   |     ( X = 2,
((S = 1,
       (((Z = 1,var(Y),Y = 900)
         | (Z = 1,nonvar(Y),Y =< 900))
      |  ((Z = 2,var(Y),Y = 1125)
          | (Z = 2,nonvar(Y),Y =< 1125))
      |  ((Z = 3,var(Y),Y = 1350)
          | (Z = 3,nonvar(Y),Y =< 1350))))
```

```
| (S = 2,
                (((Z = 1,var(Y),Y = 1800)
                   | (Z = 1,nonvar(Y),Y =< 1800))
               |  ((Z = 2,var(Y),Y = 2250)
                   | (Z = 2,nonvar(Y),Y =< 2250))
               |  ((Z = 3,var(Y),Y = 2700)
                   | (Z = 3,nonvar(Y),Y =< 2700))))))
          |     ( X = 3,
      ((S = 1,
                (((Z = 1,var(Y),Y = 600)
                   | (Z = 1,nonvar(Y),Y =< 600))
               |  ((Z = 2,var(Y),Y = 750)
                   | (Z = 2,nonvar(Y),Y =< 750))
               |  ((Z = 3,var(Y),Y = 900)
                   | (Z = 3,nonvar(Y),Y =< 900))))
| (S = 2,
                (((Z = 1,var(Y),Y = 1200)
                   | (Z = 1,nonvar(Y),Y =< 1200))
               |  ((Z = 2,var(Y),Y = 1500)
                   | (Z = 2,nonvar(Y),Y =< 1500))
               |  ((Z = 3,var(Y),Y = 1800)
                   | (Z = 3,nonvar(Y),Y =< 1800)))))))).


          rule(36,S,X,Y,Z) :-
                   xkb_identify(' Group C '),
                  (( X = 1,

       ((S = 1,
                (((Z = 1,var(Y),Y = 1800)
                   | (Z = 1,nonvar(Y),Y =< 1800))
               |  ((Z = 2,var(Y),Y = 2250)
                   | (Z = 2,nonvar(Y),Y =< 2250))
               |  ((Z = 3,var(Y),Y = 2700)
                   | (Z = 3,nonvar(Y),Y =< 2700))))
        |  (S = 2,
               (((Z = 1,var(Y),Y = 3600)
                  | (Z = 1,nonvar(Y),Y =< 3600))
               |  ((Z = 2,var(Y),Y = 4500)
                   | (Z = 2,nonvar(Y),Y =< 4500))
               |  ((Z = 3,var(Y),Y = 5400)
                   | (Z = 3,nonvar(Y),Y =< 5400))))))

          |     ( X = 2,
      ((S = 1,
                (((Z = 1,var(Y),Y = 900)
                   | (Z = 1,nonvar(Y),Y =< 900))
               |  ((Z = 2,var(Y),Y = 1125)
                   | (Z = 2,nonvar(Y),Y =< 1125))
               |  ((Z = 3,var(Y),Y = 1350)
                   | (Z = 3,nonvar(Y),Y =< 1350))))
```

```
| (S = 2,
                (((Z = 1,var(Y),Y = 1800)
                   | (Z = 1,nonvar(Y),Y =< 1800))
              |  ((Z = 2,var(Y),Y = 2250)
                   | (Z = 2,nonvar(Y),Y =< 2250))
              |  ((Z = 3,var(Y),Y = 2700)
                   | (Z = 3,nonvar(Y),Y =< 2700))))))
        |     ( X = 3,
    ((S = 1,
                (((Z = 1,var(Y),Y = 600)
                   | (Z = 1,nonvar(Y),Y =< 600))
              |  ((Z = 2,var(Y),Y = 750)
                   | (Z = 2,nonvar(Y),Y =< 750))
              |  ((Z = 3,var(Y),Y = 900)
                   | (Z = 3,nonvar(Y),Y =< 900))))
| (S = 2,
                (((Z = 1,var(Y),Y = 1200)
                   | (Z = 1,nonvar(Y),Y =< 1200))
              |  ((Z = 2,var(Y),Y = 1500)
                   | (Z = 2,nonvar(Y),Y =< 1500))
              |  ((Z = 3,var(Y),Y = 1800)
                   | (Z = 3,nonvar(Y),Y =< 1800))))))).



          rule(37,S,X,Y,Z) :-
                   xkb_identify(' Group C '),
                  (( X = 1,

      ((S = 1,
              (((Z = 1,var(Y),Y = 1800)
                 | (Z = 1,nonvar(Y),Y =< 1800))
               | ((Z = 2,var(Y),Y = 2250)
                 | (Z = 2,nonvar(Y),Y =< 2250))
               | ((Z = 3,var(Y),Y = 2700)
                 | (Z = 3,nonvar(Y),Y =< 2700))))
       |  (S = 2,
              (((Z = 1,var(Y),Y = 3600)
                 | (Z = 1,nonvar(Y),Y =< 3600))
              |  ((Z = 2,var(Y),Y = 4500)
                 | (Z = 2,nonvar(Y),Y =< 4500))
              |  ((Z = 3,var(Y),Y = 5400)
                 | (Z = 3,nonvar(Y),Y =< 5400))))))

         |     ( X = 2,
    ((S = 1,
              (((Z = 1,var(Y),Y = 900)
                 | (Z = 1,nonvar(Y),Y =< 900))
              |  ((Z = 2,var(Y),Y = 1125)
                 | (Z = 2,nonvar(Y),Y =< 1125))
              |  ((Z = 3,var(Y),Y = 1350)
```

```
                | (Z = 3,nonvar(Y),Y =< 1350))))
| (S = 2,
              (((Z = 1,var(Y),Y = 1800)
                | (Z = 1,nonvar(Y),Y =< 1800))
            |  ((Z = 2,var(Y),Y = 2250)
                | (Z = 2,nonvar(Y),Y =< 2250))
            |  ((Z = 3,var(Y),Y = 2700)
                | (Z = 3,nonvar(Y),Y =< 2700))))))
      |     ( X = 3,
   ((S = 1,
              (((Z = 1,var(Y),Y = 600)
                | (Z = 1,nonvar(Y),Y =< 600))
            |  ((Z = 2,var(Y),Y = 750)
                | (Z = 2,nonvar(Y),Y =< 750))
            |  ((Z = 3,var(Y),Y = 900)
                | (Z = 3,nonvar(Y),Y =< 900))))
| (S = 2,
              (((Z = 1,var(Y),Y = 1200)
                | (Z = 1,nonvar(Y),Y =< 1200))
            |  ((Z = 2,var(Y),Y = 1500)
                | (Z = 2,nonvar(Y),Y =< 1500))
            |  ((Z = 3,var(Y),Y = 1800)
                | (Z = 3,nonvar(Y),Y =< 1800))))))).


      rule(38,S,X,Y,Z) :-
                xkb_identify(' Group C '),
               (( X = 1,
   ((S = 1,
          (((Z = 1,var(Y),Y = 2400)
             | (Z = 1,nonvar(Y),Y =< 2400,Y > 1800))
           | ((Z = 2,var(Y),Y = 3000)
             | (Z = 2,nonvar(Y),Y =< 3000,Y > 2250))
           | ((Z = 3,var(Y),Y = 3600)
             | (Z = 3,nonvar(Y),Y =< 3600,Y > 2700))))
    |  (S = 2,
          (((Z = 1,var(Y),Y = 4800)
             | (Z = 1,nonvar(Y),Y =< 4800,Y > 3600))
           |  ((Z = 2,var(Y),Y = 6000)
             | (Z = 2,nonvar(Y),Y =< 6000,Y > 4500))
           |  ((Z = 3,var(Y),Y = 7200)
             | (Z = 3,nonvar(Y),Y =< 7200,Y > 5400))))))

      |     ( X = 2,
   ((S = 1,
          (((Z = 1,var(Y),Y = 1200)
             | (Z = 1,nonvar(Y),Y =< 1200,Y > 900))
           |  ((Z = 2,var(Y),Y = 1500)
             | (Z = 2,nonvar(Y),Y =< 1500,Y > 1125))
           |  ((Z = 3,var(Y),Y = 1800)
             | (Z = 3,nonvar(Y),Y =< 1800,Y > 1350))))
```

```
| (S = 2,
              (((Z = 1,var(Y),Y = 2400)
                 | (Z = 1,nonvar(Y),Y =< 2400,Y > 1800))
            |  ((Z = 2,var(Y),Y = 3000)
                 | (Z = 2,nonvar(Y),Y =< 3000,Y > 2250))
            |  ((Z = 3,var(Y),Y = 3600)
                 | (Z = 3,nonvar(Y),Y =< 3600,Y > 2700))))))
       |     ( X = 3,
    ((S = 1,
              (((Z = 1,var(Y),Y = 800)
                 | (Z = 1,nonvar(Y),Y =< 800,Y > 600))
            |  ((Z = 2,var(Y),Y = 1000)
                 | (Z = 2,nonvar(Y),Y =< 1000,Y > 750))
            |  ((Z = 3,var(Y),Y = 1200)
                 | (Z = 3,nonvar(Y),Y =< 1200,Y > 900))))
| (S = 2,
              (((Z = 1,var(Y),Y = 1600)
                 | (Z = 1,nonvar(Y),Y =< 1600,Y > 1200))
            |  ((Z = 2,var(Y),Y = 2000)
                 | (Z = 2,nonvar(Y),Y =< 2000,Y > 1500))
            |  ((Z = 3,var(Y),Y = .  0)
                 | (Z = 3,nonvar(Y),. =< 2400,Y >
                                      1800))))))).



        rule(39,S,X,Y,Z) :-
                 xkb_identify(' Group C '),

                 (( X = 1,

    ((S = 1,
            (((Z = 1,var(Y),Y = 2400)
               | (Z = 1,nonvar(Y),Y =< 2400,Y > 1800))
             | ((Z = 2,var(Y),Y = 3000)
               | (Z = 2,nonvar(Y),Y =< 3000,Y > 2250))
             | ((Z = 3,var(Y),Y = 3600)
               | (Z = 3,nonvar(Y),Y =< 3600,Y > 2700))))
     |  (S = 2,
           (((Z = 1,var(Y),Y = 4800)
              | (Z = 1,nonvar(Y),Y =< 4800,Y > 3600))
           |  ((Z = 2,var(Y),Y = 6000)
              | (Z = 2,nonvar(Y),Y =< 6000,Y > 4500))
           |  ((Z = 3,var(Y),Y = 7200)
              | (Z = 3,nonvar(Y),Y =< 7200,Y > 5400))))))

       |     ( X = 2,
   ((S = 1,
            (((Z = 1,var(Y),Y = 1200)
               | (Z = 1,nonvar(Y),Y =< 1200,Y > 900))
           |  ((Z = 2,var(Y),Y = 1500)
```

```
                  | (Z = 2,nonvar(Y),Y =< 1500,Y > 1125))
              |   ((Z = 3,var(Y),Y = 1800)
                  | (Z = 3,nonvar(Y),Y =< 1800,Y > 1350))))
| (S = 2,
              (((Z = 1,var(Y),Y = 2400)
                | (Z = 1,nonvar(Y),Y =< 2400,Y > 1800))
              |   ((Z = 2,var(Y),Y = 3000)
                  | (Z = 2,nonvar(Y),Y =< 3000,Y > 2250))
              |   ((Z = 3,var(Y),Y = 3600)
                  | (Z = 3,nonvar(Y),Y =< 3600,Y > 2700))))))
      |     ( X = 3,
   ((S = 1,
              (((Z = 1,var(Y),Y = 800)
                | (Z = 1,nonvar(Y),Y =< 800,Y > 600))
              |   ((Z = 2,var(Y),Y = 1000)
                  | (Z = 2,nonvar(Y),Y =< 1000,Y > 750))
              |   ((Z = 3,var(Y),Y = 1200)
                  | (Z = 3,nonvar(Y),Y =< 1200,Y > 900))))
| (S = 2,
              (((Z = 1,var(Y),Y = 1600)
                | (Z = 1,nonvar(Y),Y =< 1600,Y > 1200))
              |   ((Z = 2,var(Y),Y = 2000)
                  | (Z = 2,nonvar(Y),Y =< 2000,Y > 1500))
              |   ((Z = 3,var(Y),Y = 2400)
                  | (Z = 3,nonvar(Y),Y =< 2400,Y >
                                        1800))))))).


        rule(40,S,X,Y,Z) :-
                  xkb_identify(' Group C '),

                  (( X = 1,

    ((S = 1,
              (((Z = 1,var(Y),Y = 2400)
                | (Z = 1,nonvar(Y),Y =< 2400,Y > 1800))
              |   ((Z = 2,var(Y),Y = 3000)
                  | (Z = 2,nonvar(Y),Y =< 3000,Y > 2250))
              |   ((Z = 3,var(Y),Y = 3600)
                  | (Z = 3,nonvar(Y),Y =< 3600,Y > 2700))))
     |   (S = 2,
              (((Z = 1,var(Y),Y = 4800)
                | (Z = 1,nonvar(Y),Y =< 4800,Y > 3600))
              |   ((Z = 2,var(Y),Y = 6000)
                  | (Z = 2,nonvar(Y),Y =< 6000,Y > 4500))
              |   ((Z = 3,var(Y),Y = 7200)
                  | (Z = 3,nonvar(Y),Y =< 7200,Y > 5400))))))

      |     ( X = 2,
   ((S = 1,
              (((Z = 1,var(Y),Y = 1200)
```

```
                | (Z = 1,nonvar(Y),Y =< 1200,Y > 900))
            |  ((Z = 2,var(Y),Y = 1500)
                | (Z = 2,nonvar(Y),Y =< 1500,Y > 1125))
            |  ((Z = 3,var(Y),Y = 1800)
                | (Z = 3,nonvar(Y),Y =< 1800,Y > 1350))))
| (S = 2,
            (((Z = 1,var(Y),Y = 2400)
                | (Z = 1,nonvar(Y),Y =< 2400,Y > 1800))
            |  ((Z = 2,var(Y),Y = 3000)
                | (Z = 2,nonvar(Y),Y =< 3000,Y > 2250))
            |  ((Z = 3,var(Y),Y = 3600)
                | (Z = 3,nonvar(Y),Y =< 3600,Y > 2700))))))
      |    ( X = 3,
   ((S = 1,
            (((Z = 1,var(Y),Y = 800)
                | (Z = 1,nonvar(Y),Y =< 800,Y > 600))
            |  ((Z = 2,var(Y),Y = 1000)
                | (Z = 2,nonvar(Y),Y =< 1000,Y > 750))
            |  ((Z = 3,var(Y),Y = 1200)
                | (Z = 3,nonvar(Y),Y =< 1200,Y > 900))))
| (S = 2,
            (((Z = 1,var(Y),Y = 1600)
                | (Z = 1,nonvar(Y),Y =< 1600,Y > 1200))
            |  ((Z = 2,var(Y),Y = 2000)
                | (Z = 2,nonvar(Y),Y =< 2000,Y > 1500))
            |  ((Z = 3,var(Y),Y = 2400)
                | (Z = 3,nonvar(Y),Y =< 2400,Y >
                                        1800)))))))).


        rule(41,S,X,Y,Z) :-
                xkb_identify(' Group C '),
                X = 4,
                S = 2,
            (((Z = 1,var(Y),Y = 1200)
                | (Z = 1,nonvar(Y),Y =< 1200))
          |  ((Z = 2,var(Y),Y = 1500)
                | (Z = 2,nonvar(Y),Y =< 1500))
          |  ((Z = 3,var(Y),Y = 1800)
                | (Z = 3,nonvar(Y),Y =< 1800))).


        rule(42,S,X,Y,Z) :-
                xkb_identify(' Group C '),
                X = 4,
                S = 2,
            (((Z = 1,var(Y),Y = 1200)
                | (Z = 1,nonvar(Y),Y =< 1200))
          |  ((Z = 2,var(Y),Y = 1500)
                | (Z = 2,nonvar(Y),Y =< 1500))
```

```
        |  ((Z = 3,var(Y),Y = 1800)
            | (Z = 3,nonvar(Y),Y =< 1800))).



    rule(43,S,X,Y,Z) :-
              xkb_identify(' Group C '),
              X = 4,
              S = 2,
          (((Z = 1,var(Y),Y = 1200)
            | (Z = 1,nonvar(Y),Y =< 1200))
        |  ((Z = 2,var(Y),Y = 1500)
            | (Z = 2,nonvar(Y),Y =< 1500))
        |  ((Z = 3,var(Y),Y = 1800)
            | (Z = 3,nonvar(Y),Y =< 1800))).



    rule(44,S,X,Y,Z) :-
              xkb_identify(' Group C '),
             (( X = 1,
  ((S = 1,
        (((Z = 1,var(Y),Y = 'unlimited')
          | (Z = 1,nonvar(Y),Y > 2400))
        | ((Z = 2,var(Y),Y = 'unlimited')
          | (Z = 2,nonvar(Y),Y > 3000))
        | ((Z = 3,var(Y),Y = 'unlimited')
          | (Z = 3,nonvar(Y),Y > 3600))))
   |  (S = 2,
        (((Z = 1,var(Y),Y = 'unlimited')
          | (Z = 1,nonvar(Y),Y > 4800))
        |  ((Z = 2,var(Y),Y = 'unlimited')
            | (Z = 2,nonvar(Y),Y > 6000))
        |  ((Z = 3.var(Y),Y = 'unlimited')
            | (Z = 3,nonvar(Y),Y > 7200))))))

     |     ( X = 2,
  ((S = 1,
        (((Z = 1,var(Y),Y = 6000)
          | (Z = 1,nonvar(Y),Y =< 6000,Y > 1200))
        |  ((Z = 2,var(Y),Y = 'unlimited')
            | (Z = 2,nonvar(Y),Y > 1500))
        |  ((Z = 3,var(Y),Y = 'unlimited')
            | (Z = 3,nonvar(Y),Y > 1800))))
| (S = 2,
        (((Z = 1,var(Y),Y = 12000)
          | (Z = 1,nonvar(Y),Y =< 12000,Y > 2400))
        |  ((Z = 2,var(Y),Y = 'unlimited')
            | (Z = 2,nonvar(Y),Y > 3000))
        |  ((Z = 3,var(Y),Y = 'unlimited')
            | (Z = 3,nonvar(Y),Y > 3600))))))
```

```
        |     ( X = 3,
    ((S = 1,
                (((Z = 1,var(Y),Y = 4000)
                  | (Z = 1,nonvar(Y),Y =< 4000,Y > 800))
              |  ((Z = 2,var(Y),Y = 5000)
                  | (Z = 2,nonvar(Y),Y =< 5000,Y > 1000))
              |  ((Z = 3,var(Y),Y = 6000)
                  | (Z = 3,nonvar(Y),Y =< 6000,Y > 1200))))
| (S = 2,
                (((Z = 1,var(Y),Y = 8000)
                  | (Z = 1,nonvar(Y),Y =< 8000,Y > 1600))
              |  ((Z = 2,var(Y),Y = 10000)
                  | (Z = 2,nonvar(Y),Y =< 10000,Y > 2000))
              |  ((Z = 3,var(Y),Y = 12000)
                  | (Z = 3,nonvar(Y),Y =< 12000,Y >
                                          2400))))))

        |     ( X = 4,
     ((S = 1,
              (((Z = 1,var(Y),Y = 3000)
                 | (Z = 1,nonvar(Y),Y =< 3000))
               | ((Z = 2,var(Y),Y = 3750)
                 | (Z = 2,nonvar(Y),Y =< 3750))
               | ((Z = 3,var(Y),Y = 4500)
                 | (Z = 3,nonvar(Y),Y =< 4500))))
      |  (S = 2,
            (((Z = 1,var(Y),Y = 6000)
               | (Z = 1,nonvar(Y),Y =< 6000,Y > 1200))
            |  ((Z = 2,var(Y),Y = 7500)
               | (Z = 2,nonvar(Y),Y =< 7500,Y > 1500))
            |  ((Z = 3,var(Y),Y = 9000)
               | (Z = 3,nonvar(Y),Y =< 9000,Y > 1800))))))

        |     ( X = 5,
    ((S = 1,
              (((Z = 1,var(Y),Y = 2400)
                 | (Z = 1,nonvar(Y),Y =< 2400))
            |  ((Z = 2,var(Y),Y = 3000)
                 | (Z = 2,nonvar(Y),Y =< 3000))
            |  ((Z = 3,var(Y),Y = 3600)
                 | (Z = 3,nonvar(Y),Y =< 3600))))
| (S = 2,
              (((Z = 1,var(Y),Y = 4800)
                 | (Z = 1,nonvar(Y),Y =< 4800))
            |  ((Z = 2,var(Y),Y = 6000)
                 | (Z = 2,nonvar(Y),Y =< 6000))
            |  ((Z = 3,var(Y),Y = 7200)
                 | (Z = 3,nonvar(Y),Y =< 7200))))))
        |     ( X = 6,
    ((S = 1,
              (((Z = 1,var(Y),Y = 2000)
                 | (Z = 1,nonvar(Y),Y =< 2000))
```

```
            |  ((Z = 2,var(Y),Y = 2500)
                | (Z = 2,nonvar(Y),Y =< 2500))
            |  ((Z = 3,var(Y),Y = 3000)
                | (Z = 3,nonvar(Y),Y =< 3000))))
| (S = 2,
            (((Z = 1,var(Y),Y = 4000)
                | (Z = 1,nonvar(Y),Y =< 4000))
            |  ((Z = 2,var(Y),Y = 5000)
                | (Z = 2,nonvar(Y),Y =< 5000))
            |  ((Z = 3,var(Y),Y = 6000)
                | (Z = 3,nonvar(Y),Y =< 6000))))))).



    rule(45,S,X,Y,Z) :-
                xkb_identify(' Group C '),

                (( X = 1,

    ((S = 1,
            (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 2400))
            | ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 3000))
            | ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 3600))))
     |  (S = 2,
            (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 4800))
            |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 6000))
            |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 7200))))))

      |     ( X = 2,
   ((S = 1,
            (((Z = 1,var(Y),Y = 6000)
                | (Z = 1,nonvar(Y),Y =< 6000,Y > 1200))
            |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 1500))
            |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 1800))))
| (S = 2,
            (((Z = 1,var(Y),Y = 12000)
                | (Z = 1,nonvar(Y),Y =< 12000,Y > 2400))
            |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 3000))
            |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 3600))))))
      |     ( X = 3,
   ((S = 1,
```

```
                (((Z = 1,var(Y),Y = 4000)
                  | (Z = 1,nonvar(Y),Y =< 4000,Y > 800))
              |  ((Z = 2,var(Y),Y = 5000)
                  | (Z = 2,nonvar(Y),Y =< 5000,Y > 1000))
              |  ((Z = 3,var(Y),Y = 6000)
                  | (Z = 3,nonvar(Y),Y =< 6000,Y > 1200))))
| (S = 2,
                (((Z = 1,var(Y),Y = 8000)
                  | (Z = 1,nonvar(Y),Y =< 8000,Y > 1600))
              |  ((Z = 2,var(Y),Y = 10000)
                  | (Z = 2,nonvar(Y),Y =< 10000,Y > 2000))
              |  ((Z = 3,var(Y),Y = 12000)
                  | (Z = 3,nonvar(Y),Y =< 12000,Y >
                                          2400))))))

        |      ( X = 4,
     ((S = 1,
              (((Z = 1,var(Y),Y = 3000)
                  | (Z = 1,nonvar(Y),Y =< 3000))
               | ((Z = 2,var(Y),Y = 3750)
                  | (Z = 2,nonvar(Y),Y =< 3750))
               | ((Z = 3,var(Y),Y = 4500)
                  | (Z = 3,nonvar(Y),Y =< 4500))))
      |  (S = 2,
             (((Z = 1,var(Y),Y = 6000)
                | (Z = 1,nonvar(Y),Y =< 6000,Y > 1200))
              |  ((Z = 2,var(Y),Y = 7500)
                  | (Z = 2,nonvar(Y),Y =< 7500,Y > 1500))
              |  ((Z = 3,var(Y),Y = 9000)
                  | (Z = 3,nonvar(Y),Y =< 9000,Y > 1800))))))

        |     ( X = 5,
    ((S = 1,
              (((Z = 1,var(Y),Y = 2400)
                  | (Z = 1,nonvar(Y),Y =< 2400))
              |  ((Z = 2,var(Y),Y = 3000)
                  | (Z = 2,nonvar(Y),Y =< 3000))
              |  ((Z = 3,var(Y),Y = 3600)
                  | (Z = 3,nonvar(Y),Y =< 3600))))
| (S = 2,
                (((Z = 1,var(Y),Y = 4800)
                   | (Z = 1,nonvar(Y),Y =< 4800))
              |  ((Z = 2,var(Y),Y = 6000)
                  | (Z = 2,nonvar(Y),Y =< 6000))
              |  ((Z = 3,var(Y),Y = 7200)
                  | (Z = 3,nonvar(Y),Y =< 7200))))))
        |     ( X = 6,
    ((S = 1,
              (((Z = 1,var(Y),Y = 2000)
                  | (Z = 1,nonvar(Y),Y =< 2000))
              |  ((Z = 2,var(Y),Y = 2500)
                  | (Z = 2,nonvar(Y),Y =< 2500))
```

```
                | ((Z = 3,var(Y),Y = 3000)
                  | (Z = 3,nonvar(Y),Y =< 3000))))
| (S = 2,
              (((Z = 1,var(Y),Y = 4000)
                | (Z = 1,nonvar(Y),Y =< 4000))
              | ((Z = 2,var(Y),Y = 5000)
                | (Z = 2,nonvar(Y),Y =< 5000))
              | ((Z = 3,var(Y),Y = 6000)
                | (Z = 3,nonvar(Y),Y =< 6000))))))).


        rule(46,S,X,Y,Z) :-
                xkb_identify(' Group C '),

                  (( X = 2,
    ((S = 1,
              ((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 6000)))
| (S = 2,
              ((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 12000)))))
      |     ( X = 3,
    ((S = 1,
              (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 4000))
              | ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 5000))
              | ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 6000))))
| (S = 2,
              (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 8000))
              | ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 10000))
              | ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 12000)))))))

      |       ( X = 4,
     ((S = 1,
             (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 3000))
             | ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 3750))
             | ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 4500))))
      | (S = 2,
           (((Z = 1,var(Y),Y = 'unlimited')
              | (Z = 1,nonvar(Y),Y > 6000))
           | ((Z = 2,var(Y),Y = 'unlimited')
              | (Z = 2,nonvar(Y),Y > 7500))
           | ((Z = 3,var(Y),Y = 'unlimited')
              | (Z = 3,nonvar(Y),Y > 9000)))))))
```

```
        |     ( X = 5,
    ((S = 1,
              (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 2400))
             |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 3000))
             |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 3600))))
| (S = 2,
              (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 4800))
             |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 6000))
             |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 7200))))))
        |     ( X = 6,
    ((S = 1,
              (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 2000))
             |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 2500))
             |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 3000))))
| (S = 2,
              (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 4000))
             |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 5000))
             |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 6000))))))
        |     ( nonvar(X),X > 6)).



        rule(47,S,X,Y,Z) :-
                xkb_identify(' Group C '),


                (( X = 2,
    ((S = 1,
              ((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 6000)))
| (S = 2,
              ((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 12000)))))
        |     ( X = 3,
    ((S = 1,
              (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 4000))
             |  ((Z = 2,var(Y),Y = 'unlimited')
```

```
                | (Z = 2,nonvar(Y),Y > 5000))
            |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 6000))))
| (S = 2,
            (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 8000))
            |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 10000))
            |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 12000)))))

      |      ( X = 4,
    ((S = 1,
            (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 3000))
            | ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 3750))
            | ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 4500))))
     |  (S = 2,
            (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 6000))
            |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 7500))
            |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 9000)))))

      |     ( X = 5,
   ((S = 1,
            (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 2400))
            |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 3000))
            |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 3600))))
| (S = 2,
            (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 4800))
            |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 6000))
            |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 7200)))))
      |     ( X = 6,
   ((S = 1,
            (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 2000))
            |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 2500))
            |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 3000))))
| (S = 2,
            (((Z = 1,var(Y),Y = 'unlimited')
```

```
          | (Z = 1,nonvar(Y),Y > 4000))
      |  ((Z = 2,var(Y),Y = 'unlimited')
          | (Z = 2,nonvar(Y),Y > 5000))
      |  ((Z = 3,var(Y),Y = 'unlimited')
          | (Z = 3,nonvar(Y),Y > 6000))))))
  |    ( nonvar(X),X > 6)).



        rule(48,S,X,Y,Z) :-
                  xkb_identify(' Group D '),

              (( X = 1,
((S = 1,
       (((Z = 1,var(Y),Y = 1000)
         | (Z = 1,nonvar(Y),Y =< 1000))
      | ((Z = 2,var(Y),Y = 1250)
         | (Z = 2,nonvar(Y),Y =< 1250))
      | ((Z = 3,var(Y),Y = 1500)
         | (Z = 3,nonvar(Y),Y =< 1500))))
  | (S = 2,
       (((Z = 1,var(Y),Y = 2000)
         | (Z = 1,nonvar(Y),Y =< 2000))
      |  ((Z = 2,var(Y),Y = 2500)
         | (Z = 2,nonvar(Y),Y =< 2500))
      |  ((Z = 3,var(Y),Y = 3000)
         | (Z = 3,nonvar(Y),Y =< 3000))))))

      |           (  X = 2,
((S = 1,
       (((Z = 1,var(Y),Y = 800)
         | (Z = 1,nonvar(Y),Y =< 800))
      |  ((Z = 2,var(Y),Y = 1000)
         | (Z = 2,nonvar(Y),Y =< 1000))
      |  ((Z = 3,var(Y),Y = 1200)
         | (Z = 3,nonvar(Y),Y =< 1200))))
  | (S = 2,
       (((Z = 1,var(Y),Y = 1600)
         | (Z = 1,nonvar(Y),Y =< 1600))
      |  ((Z = 2,var(Y),Y = 2000)
         | (Z = 2,nonvar(Y),Y =< 2000))
      |  ((Z = 3,var(Y),Y = 2400)
         | (Z = 3,nonvar(Y),Y =< 2400))))))).


    rule(49,S,X,Y,Z) :-
              xkb_identify(' Group D '),

            (( X = 1,
```

```
      ((S = 1,
              (((Z = 1,var(Y),Y = 4800)
                 | (Z = 1,nonvar(Y),Y =< 4800,Y > 1000))
               | ((Z = 2,var(Y),Y = 6000)
                  | (Z = 2,nonvar(Y),Y =< 6000,Y > 1250))
               | ((Z = 3,var(Y),Y = 7200)
                  | (Z = 3,nonvar(Y),Y =< 7200,Y > 1500))))
       |  (S = 2,
             (((Z = 1,var(Y),Y = 9600)
                | (Z = 1,nonvar(Y),Y =< 9600,Y > 2000))
             |  ((Z = 2,var(Y),Y = 12000)
                 | (Z = 2,nonvar(Y),Y =< 12000,Y > 2500))
             |  ((Z = 3,var(Y),Y = 14400)
                 | (Z = 3,nonvar(Y),Y =< 14400,Y >
                                          3000))))))

         |     ( X = 2,
    ((S = 1,
              (((Z = 1,var(Y),Y = 2400)
                | (Z = 1,nonvar(Y),Y =< 2400,Y > 800))
             |  ((Z = 2,var(Y),Y = 3000)
                 | (Z = 2,nonvar(Y),Y =< 3000,Y > 1000))
             |  ((Z = 3,var(Y),Y = 3600)
                 | (Z = 3,nonvar(Y),Y =< 3600,Y > 1200))))
| (S = 2,
              (((Z = 1,var(Y),Y = 4800)
                 | (Z = 1,nonvar(Y),Y =< 4800,Y > 1600))
             |  ((Z = 2,var(Y),Y = 6000)
                 | (Z = 2,nonvar(Y),Y =< 6000,Y > 2000))
             |  ((Z = 3,var(Y),Y = 7200)
                 | (Z = 3,nonvar(Y),Y =< 7200,Y > 2400))))))
         |     ( X = 3,
    ((S = 1,
              (((Z = 1,var(Y),Y = 1600)
                | (Z = 1,nonvar(Y),Y =< 1600))
             |  ((Z = 2,var(Y),Y = 2000)
                 | (Z = 2,nonvar(Y),Y =< 2000))
             |  ((Z = 3,var(Y),Y = 2400)
                 | (Z = 3,nonvar(Y),Y =< 2400))))
| (S = 2,
              (((Z = 1,var(Y),Y = 3200)
                | (Z = 1,nonvar(Y),Y =< 3200))
             |  ((Z = 2,var(Y),Y = 4000)
                 | (Z = 2,nonvar(Y),Y =< 4000))
             |  ((Z = 3,var(Y),Y = 4800)
                 | (Z = 3,nonvar(Y),Y =< 4800))))))).



rule(50,S,X,Y,Z) :-
                    xkb_identify(' Group C   '),
```

```
                X = 1,

      ((S = 1,
            (((Z = 1,var(Y),Y = 2400)
             | (Z = 1,nonvar(Y), Y =< 2400))
          |  ((Z = 2,var(Y),Y =3000)
             |(Z = 2,nonvar(Y),Y =< 3000))
          |  ((Z = 3,var(Y),Y = 3600)
             |(Z = 3,nonvar(Y),Y =< 3600 ))))
        | (S = 2,
            (((Z = 1,var(Y),Y = 4800)
             | (Z = 1,nonvar(Y),Y =< 4800))
          |  ((Z = 2,var(Y),Y = 6000)
             |(Z = 2,nonvar(Y),Y =< 6000))
          |  ((Z = 3,var(Y),Y = 7200)
             |(Z = 3,nonvar(Y),Y =< 7200))))).


    rule(51,S,X,Y,Z) :-
              xkb_identify(' Group D '),

              (( X = 1,

     ((S = 1,
          (((Z = 1,var(Y),Y = 'unlimited')
            | (Z = 1,nonvar(Y),Y > 4800))
          | ((Z = 2,var(Y),Y = 'unlimited')
            | (Z = 2,nonvar(Y),Y > 6000))
          | ((Z = 3,var(Y),Y = 'unlimited')
            | (Z = 3,nonvar(Y),Y > 7200))))
      |  (S = 2,
          (((Z = 1,var(Y),Y = 'unlimited')
            | (Z = 1,nonvar(Y),Y > 9600))
          |  ((Z = 2,var(Y),Y = 'unlimited')
             | (Z = 2,nonvar(Y),Y > 12000))
          |  ((Z = 3,var(Y),Y = 'unlimited')
             | (Z = 3,nonvar(Y),Y > 14400))))))

        |     ( X = 2,
   ((S = 1,
          (((Z = 1,var(Y),Y = 7200)
            | (Z = 1,nonvar(Y),Y =< 7200,Y > 2400))
          |  ((Z = 2,var(Y),Y = 'unlimited')
             | (Z = 2,nonvar(Y),Y > 3000))
          |  ((Z = 3,var(Y),Y = 'unlimited')
             | (Z = 3,nonvar(Y),Y > 3600))))
| (S = 2,
          (((Z = 1,var(Y),Y = 14400)
            | (Z = 1,nonvar(Y),Y =< 14400,Y > 4800))
          |  ((Z = 2,var(Y),Y = 'unlimited')
             | (Z = 2,nonvar(Y),Y > 6000))
```

```
                | ((Z = 3,var(Y),Y = 'unlimited')
                  | (Z = 3,nonvar(Y),Y > 7200))))))
          |    ( X = 3,
    ((S = 1,
              (((Z = 1,var(Y),Y = 4800)
                | (Z = 1,nonvar(Y),Y =< 4800,Y > 1600))
              | ((Z = 2,var(Y),Y = 6000)
                | (Z = 2,nonvar(Y),Y =< 6000,Y > 2000))
              | ((Z = 3,var(Y),Y = 7200)
                | (Z = 3,nonvar(Y),Y =< 7200,Y > 2400))))
| (S = 2,
              (((Z = 1,var(Y),Y = 9600)
                | (Z = 1,nonvar(Y),Y =< 9600,Y > 3200))
              | ((Z = 2,var(Y),Y = 12000)
                | (Z = 2,nonvar(Y),Y =< 12000,Y > 4000))
              | ((Z = 3,var(Y),Y = 14400)
                | (Z = 3,nonvar(Y),Y =< 14400,Y >
                                              4800))))))

          |    ( X = 4,
     ((S = 1,
             (((Z = 1,var(Y),Y = 3600)
                | (Z = 1,nonvar(Y),Y =< 3600))
             | ((Z = 2,var(Y),Y = 4500)
                | (Z = 2,nonvar(Y),Y =< 4500))
             | ((Z = 3,var(Y),Y = 5400)
                | (Z = 3,nonvar(Y),Y =< 5400))))
      | (S = 2,
             (((Z = 1,var(Y),Y = 7200)
                | (Z = 1,nonvar(Y),Y =< 7200))
             | ((Z = 2,var(Y),Y = 9000)
                | (Z = 2,nonvar(Y),Y =< 9000))
             | ((Z = 3,var(Y),Y = 10800)
                | (Z = 3,nonvar(Y),Y =< 10800))))))

          |    ( X = 5,
    ((S = 1,
              (((Z = 1,var(Y),Y = 2800)
                | (Z = 1,nonvar(Y),Y =< 2800))
              | ((Z = 2,var(Y),Y = 3600)
                | (Z = 2,nonvar(Y),Y =< 3600))
              | ((Z = 3,var(Y),Y = 4320)
                | (Z = 3,nonvar(Y),Y =< 4320))))
| (S = 2,
              (((Z = 1,var(Y),Y = 5600)
                 | (Z = 1,nonvar(Y),Y =< 5600))
              | ((Z = 2,var(Y),Y = 7200)
                | (Z = 2,nonvar(Y),Y =< 7200))
              | ((Z = 3,var(Y),Y = 8640)
                | (Z = 3,nonvar(Y),Y =< 8640))))))
          |    ( X = 6,
    ((S = 1,
```

```
                (((Z = 1,var(Y),Y = 2400)
                  | (Z = 1,nonvar(Y),Y =< 2400))
              |  ((Z = 2,var(Y),Y = 3000)
                  | (Z = 2,nonvar(Y),Y =< 3000))
              |  ((Z = 3,var(Y),Y = 3600)
                  | (Z = 3,nonvar(Y),Y =< 3600))))
| (S = 2,
                (((Z = 1,var(Y),Y = 4800)
                  | (Z = 1,nonvar(Y),Y =< 4800))
              |  ((Z = 2,var(Y),Y = 6000)
                  | (Z = 2,nonvar(Y),Y =< 6000))
              |  ((Z = 3,var(Y),Y = 7200)
                  | (Z = 3,nonvar(Y),Y =< 7200))))))).


                    rule(52,S,X,Y,Z) :-
                            xkb_identify(' Group D '),

                     (( X = 2,
    ((S = 1,
                ((Z = 1,var(Y),Y = 'unlimited')
                  | (Z = 1,nonvar(Y),Y > 7200)))
| (S = 2,
                ((Z = 1,var(Y),Y = 'unlimited')
                   | (Z = 1,nonvar(Y),Y > 144000)))))
        |     ( X = 3,
    ((S = 1,
                (((Z = 1,var(Y),Y = 'unlimited')
                  | (Z = 1,nonvar(Y),Y > 4800))
              |  ((Z = 2,var(Y),Y = 'unlimited')
                  | (Z = 2,nonvar(Y),Y > 6000))
              |  ((Z = 3,var(Y),Y = 'unlimited')
                  | (Z = 3,nonvar(Y),Y > 7200))))
| (S = 2,
                (((Z = 1,var(Y),Y = 'unlimited')
                  | (Z = 1,nonvar(Y),Y > 9600))
              |  ((Z = 2,var(Y),Y = 'unlimited')
                  | (Z = 2,nonvar(Y),Y > 12000))
              |  ((Z = 3,var(Y),Y = 'unlimited')
                  | (Z = 3,nonvar(Y),Y > 14400)))))))

          |      ( X = 4,
     ((S = 1,
              (((Z = 1,var(Y),Y = 'unlimited')
                  | (Z = 1,nonvar(Y),Y > 3600))
             | ((Z = 2,var(Y),Y = 'unlimited')
                  | (Z = 2,nonvar(Y),Y > 4500))
             | ((Z = 3,var(Y),Y = 'unlimited')
                  | (Z = 3,nonvar(Y),Y > 5400))))
      |   (S = 2,
             (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 7200))
```

```
                |   ((Z = 2,var(Y),Y = 'unlimited')
                    | (Z = 2,nonvar(Y),Y > 9000))
                |   ((Z = 3,var(Y),Y = 'unlimited')
                    | (Z = 3,nonvar(Y),Y > 10800))))))

        |     ( X = 5,
    ((S = 1,
                  (((Z = 1,var(Y),Y = 'unlimited')
                    | (Z = 1,nonvar(Y),Y > 2800))
                |   ((Z = 2,var(Y),Y = 'unlimited')
                    | (Z = 2,nonvar(Y),Y > 3600))
                |   ((Z = 3,var(Y),Y = 'unlimited')
                    | (Z = 3,nonvar(Y),Y > 4320))))
| (S = 2,
                  (((Z = 1,var(Y),Y = 'unlimited')
                     | (Z = 1,nonvar(Y),Y > 5600))
                |   ((Z = 2,var(Y),Y = 'unlimited')
                    | (Z = 2,nonvar(Y),Y > 7200))
                |   ((Z = 3,var(Y),Y = 'unlimited')
                    | (Z = 3,nonvar(Y),Y > 8640))))))
        |     ( X = 6,
    ((S = 1,
                  (((Z = 1,var(Y),Y = 'unlimited')
                    | (Z = 1,nonvar(Y),Y > 2400))
                |   ((Z = 2,var(Y),Y = 'unlimited')
                    | (Z = 2,nonvar(Y),Y > 3000))
                |   ((Z = 3,var(Y),Y = 'unlimited')
                    | (Z = 3,nonvar(Y),Y > 3600))))
| (S = 2,
                  (((Z = 1,var(Y),Y = 'unlimited')
                    | (Z = 1,nonvar(Y),Y > 4800))
                |   ((Z = 2,var(Y),Y = 'unlimited')
                    | (Z = 2,nonvar(Y),Y > 6000))
                |   ((Z = 3,var(Y),Y = 'unlimited')
                    | (Z = 3,nonvar(Y),Y > 7200))))))
        |     ( nonvar(X),X > 6)).


                rule(53,S,X,Y,Z) :-
                        xkb_identify(' Group E '),

                    (( X = 1,
      ((S = 1,
             (((Z = 1,var(Y),Y = 1000)
               | (Z = 1,nonvar(Y),Y =< 1000))
             | ((Z = 2,var(Y),Y = 1250)
               | (Z = 2,nonvar(Y),Y =< 1250))
             | ((Z = 3,var(Y),Y = 1500)
               | (Z = 3,nonvar(Y),Y =< 1500))))
        | (S = 2,
             (((Z = 1,var(Y),Y = 2000)
               | (Z = 1,nonvar(Y),Y =< 2000))
```

```
                | ((Z = 2,var(Y),Y = 2500)
                   | (Z = 2,nonvar(Y),Y =< 2500))
                | ((Z = 3,var(Y),Y = 3000)
                   | (Z = 3,nonvar(Y),Y =< 3000))))))

                |            ( X = 2,
          ((S = 1,
                  (((Z = 1,var(Y),Y = 600)
                   | (Z = 1,nonvar(Y),Y =< 600))
                | ((Z = 2,var(Y),Y = 750)
                   | (Z = 2,nonvar(Y),Y =< 750))
                | ((Z = 3,var(Y),Y = 900)
                   | (Z = 3,nonvar(Y),Y =< 900))))
            | (S = 2,
                  (((Z = 1,var(Y),Y = 1200)
                   | (Z = 1,nonvar(Y),Y =< 1200))
                | ((Z = 2,var(Y),Y = 1500)
                   | (Z = 2,nonvar(Y),Y =< 1500))
                | ((Z = 3,var(Y),Y = 1800)
                   | (Z = 3,nonvar(Y),Y =< 1800)))))))).


        rule(54,S,X,Y,Z) :-
                  xkb_identify(' Group E '),

                  (( X = 1,

         ((S = 1,
                 (((Z = 1,var(Y),Y = 1500)
                   | (Z = 1,nonvar(Y),Y =< 1500,Y > 1000))
                 | ((Z = 2,var(Y),Y = 1500)
                   | (Z = 2,nonvar(Y),Y =< 1500,Y > 1250))
                 | ((Z = 3,var(Y),Y = 1500)
                   | (Z = 3,nonvar(Y),Y =< 1500,Y > 1500))))
          |  (S = 2,
                 (((Z = 1,var(Y),Y = 4800)
                   | (Z = 1,nonvar(Y),Y =< 4800,Y > 2000))
                | ((Z = 2,var(Y),Y = 6000)
                   | (Z = 2,nonvar(Y),Y =< 6000,Y > 2500))
                | ((Z = 3,var(Y),Y = 7200)
                   | (Z = 3,nonvar(Y),Y =< 7200,Y > 3000))))))

            |     ( X = 2,
      ((S = 1,
                  (((Z = 1,var(Y),Y = 1200)
                   | (Z = 1,nonvar(Y),Y =< 1200,Y > 600))
                | ((Z = 2,var(Y),Y = 1500)
                   | (Z = 2,nonvar(Y),Y =< 1500,Y > 750))
                | ((Z = 3,var(Y),Y = 1500)
                   | (Z = 3,nonvar(Y),Y =< 1500,Y > 900))))
| (S = 2,
```

```
                (((Z = 1,var(Y),Y = 2400)
                   | (Z = 1,nonvar(Y),Y =< 2400,Y > 1200))
              |  ((Z = 2,var(Y),Y = 3000)
                   | (Z = 2,nonvar(Y),Y =< 3000,Y > 1500))
              |  ((Z = 3,var(Y),Y = 3600)
                   | (Z = 3,nonvar(Y),Y =< 3600,Y > 1800))))))
        |     ( X = 3,
    ((S = 1,
                (((Z = 1,var(Y),Y = 800)
                   | (Z = 1,nonvar(Y),Y =< 800))
              |  ((Z = 2,var(Y),Y = 1000)
                   | (Z = 2,nonvar(Y),Y =< 1000))
              |  ((Z = 3,var(Y),Y = 1200)
                   | (Z = 3,nonvar(Y),Y =< 1200))))
| (S = 2,
                (((Z = 1,var(Y),Y = 1600)
                   | (Z = 1,nonvar(Y),Y =< 1600))
              |  ((Z = 2,var(Y),Y = 2000)
                   | (Z = 2,nonvar(Y),Y =< 2000))
              |  ((Z = 3,var(Y),Y = 2400)
                   | (Z = 3,nonvar(Y),Y =< 2400))))))).


rule(55,S,X,Y,Z) :-
                    xkb_identify(' Group E  '),
                    X = 1,

     ((S = 1,
                (((Z = 1,var(Y),Y = 2400)
                   | (Z = 1,nonvar(Y), Y =< 2400))
              |  ((Z = 2,var(Y),Y =3000)
                   |(Z = 2,nonvar(Y),Y =< 3000))
              |  ((Z = 3,var(Y),Y = 3600)
                   |(Z = 3,nonvar(Y),Y =< 3600 ))))
        | (S = 2,
                (((Z = 1,var(Y),Y = 4800)
                   | (Z = 1,nonvar(Y),Y =< 4800))
              |  ((Z = 2,var(Y),Y = 6000)
                   |(Z = 2,nonvar(Y),Y =< 6000))
              |  ((Z = 3,var(Y),Y = 7200)
                   |(Z = 3,nonvar(Y),Y =< 7200))))).


                    rule(56,S,X,Y,Z) :-
                              xkb_identify(' Group E '),

                   (( X = 1,

     ((S = 1,
              (((Z = 1,var(Y),Y = 1500)
```

```
                    | (Z = 1,nonvar(Y),Y =< 1500))
                 | ((Z = 2,var(Y),Y = 1500)
                    | (Z = 2,nonvar(Y),Y =< 1500))
                 | ((Z = 3,var(Y),Y = 1500)
                    | (Z = 3,nonvar(Y),Y =< 1500))))
       |  (S = 2,
             (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 4800))
             |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 6000))
             |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 7200))))))

     |     ( X = 2,
   ((S = 1,
             (((Z = 1,var(Y),Y = 1500)
               | (Z = 1,nonvar(Y),Y =< 1500,Y > 1200))
             |  ((Z = 2,var(Y),Y = 1500)
               | (Z = 2,nonvar(Y),Y =< 1500,Y > 1500))
             |  ((Z = 3,var(Y),Y = 1500)
               | (Z = 3,nonvar(Y),Y =< 1500,Y > 1500))))
| (S = 2,
             (((Z = 1,var(Y),Y = 7500)
                | (Z = 1,nonvar(Y),Y =< 7500,Y > 2400))
             |  ((Z = 2,var(Y),Y = 'unlimited')
               | (Z = 2,nonvar(Y),Y > 3000))
             |  ((Z = 3,var(Y),Y = 'unlimited')
               | (Z = 3,nonvar(Y),Y > 3600))))))
     |     ( X = 3,
   ((S = 1,
             (((Z = 1,var(Y),Y = 1500)
               | (Z = 1,nonvar(Y),Y =< 1500,Y > 800))
             |  ((Z = 2,var(Y),Y = 1500)
               | (Z = 2,nonvar(Y),Y =< 1500,Y > 1000))
             |  ((Z = 3,var(Y),Y = 1500)
               | (Z = 3,nonvar(Y),Y =< 1500,Y > 1200))))
| (S = 2,
             (((Z = 1,var(Y),Y = 5000)
               | (Z = 1,nonvar(Y),Y =< 5000,Y > 1600))
             |  ((Z = 2,var(Y),Y = 6250)
               | (Z = 2,nonvar(Y),Y =< 6250,Y > 2000))
             |  ((Z = 3,var(Y),Y = 7500)
               | (Z = 3,nonvar(Y),Y =< 7500,Y > 2400))))))

      |      ( X = 4,
     (S = 2,
         (((Z = 1,var(Y),Y = 3750)
            | (Z = 1,nonvar(Y),Y =< 3750))
         |  ((Z = 2,var(Y),Y = 4688)
            | (Z = 2,nonvar(Y),Y =< 4688))
         |  ((Z = 3,var(Y),Y = 5625)
            | (Z = 3,nonvar(Y),Y =< 5625)))))
```

```
        |     ( X = 5,
(S = 2,
              (((Z = 1,var(Y),Y = 3000)
                 | (Z = 1,nonvar(Y),Y =< 3000))
             |  ((Z = 2,var(Y),Y = 3750)
                 | (Z = 2,nonvar(Y),Y =< 3750))
             |  ((Z = 3,var(Y),Y = 4500)
                 | (Z = 3,nonvar(Y),Y =< 4500)))))
        |     ( X = 6,
(S = 2,
              (((Z = 1,var(Y),Y = 2500)
                 | (Z = 1,nonvar(Y),Y =< 2500))
             |  ((Z = 2,var(Y),Y = 3125)
                 | (Z = 2,nonvar(Y),Y =< 3125))
             |  ((Z = 3,var(Y),Y = 3750)
                 | (Z = 3,nonvar(Y),Y =< 3750))))))).


            rule(57,S,X,Y,Z) :-
                    xkb_identify(' Group E '),
                    S = 2,
                  (( X = 2,
              ((Z = 1,var(Y),Y = 'unlimited')
                 | (Z = 1,nonvar(Y),Y > 7500)))
        |     ( X = 3,
              (((Z = 1,var(Y),Y = 'unlimited')
                 | (Z = 1,nonvar(Y),Y > 5000))
             |  ((Z = 2,var(Y),Y = 'unlimited')
                 | (Z = 2,nonvar(Y),Y > 6250))
             |  ((Z = 3,var(Y),Y = 'unlimited')
                 | (Z = 3,nonvar(Y),Y > 7500))))

         |      ( X = 4,
            (((Z = 1,var(Y),Y = 'unlimited')
               | (Z = 1,nonvar(Y),Y > 3750))
             |  ((Z = 2,var(Y),Y = 'unlimited')
                 | (Z = 2,nonvar(Y),Y > 4688))
             |  ((Z = 3,var(Y),Y = 'unlimited')
                 | (Z = 3,nonvar(Y),Y > 5625))))

        |     ( X = 5,
              (((Z = 1,var(Y),Y = 'unlimited')
                 | (Z = 1,nonvar(Y),Y > 3000))
             |  ((Z = 2,var(Y),Y = 'unlimited')
                 | (Z = 2,nonvar(Y),Y > 3750))
             |  ((Z = 3,var(Y),Y = 'unlimited')
                 | (Z = 3,nonvar(Y),Y > 4500))))
        |     ( X = 6,
              (((Z = 1,var(Y),Y = 'unlimited')
                 | (Z = 1,nonvar(Y),Y > 2500))
             |  ((Z = 2,var(Y),Y = 'unlimited')
                 | (Z = 2,nonvar(Y),Y > 3125))
```

```
      |  ((Z = 3,var(Y),Y = 'unlimited')
          | (Z = 3,nonvar(Y),Y > 3750))))
  |    ( nonvar(X),X > 6)).


        rule(58,S,X,Y,Z) :-
                xkb_identify(' Group F Division 1 '),

             (( X = 1,
((S = 1,
       (((Z = 1,var(Y),Y = 800)
          | (Z = 1,nonvar(Y),Y =< 800))
       | ((Z = 2,var(Y),Y = 1000)
          | (Z = 2,nonvar(Y),Y =< 1000))
       | ((Z = 3,var(Y),Y = 1200)
          | (Z = 3,nonvar(Y),Y =< 1200))))
  | (S = 2,
       (((Z = 1,var(Y),Y = 1600)
          | (Z = 1,nonvar(Y),Y =< 1600))
       |  ((Z = 2,var(Y),Y = 2000)
          | (Z = 2,nonvar(Y),Y =< 2000))
       |  ((Z = 3,var(Y),Y = 2400)
          | (Z = 3,nonvar(Y),Y =< 2400))))))

       |           (  X = 2,
((S = 1,
       (((Z = 1,var(Y),Y = 400)
          | (Z = 1,nonvar(Y),Y =< 400))
       |  ((Z = 2,var(Y),Y = 500)
          | (Z = 2,nonvar(Y),Y =< 500))
       |  ((Z = 3,var(Y),Y = 600)
          | (Z = 3,nonvar(Y),Y =< 600))))
  | (S = 2,
       (((Z = 1,var(Y),Y = 800)
          | (Z = 1,nonvar(Y),Y =< 800))
       |  ((Z = 2,var(Y),Y = 1000)
          | (Z = 2,nonvar(Y),Y =< 1000))
       |  ((Z = 3,var(Y),Y = 1200)
          | (Z = 3,nonvar(Y),Y =< 1200))))))).


              rule(59,S,X,Y,Z) :-
            xkb_identify(' Group F Division 1 '),
                        S = 2,
          (( X = 1,

      (((Z = 1,var(Y),Y = 2400)
         | (Z = 1,nonvar(Y),Y =< 2400,Y > 1600))
      |  ((Z = 2,var(Y),Y = 3000)
          | (Z = 2,nonvar(Y),Y =< 3000,Y > 2000))
      |  ((Z = 3,var(Y),Y = 3600)
          | (Z = 3,nonvar(Y),Y =< 3600,Y > 2400))))
```

```
        |     ( X = 2,
                (((Z = 1,var(Y),Y = 1200)
                   | (Z = 1,nonvar(Y),Y =< 1200,Y > 800))
               |  ((Z = 2,var(Y),Y = 1500)
                   | (Z = 2,nonvar(Y),Y =< 1500,Y > 1000))
               |  ((Z = 3,var(Y),Y = 1800)
                   | (Z = 3,nonvar(Y),Y =< 1800,Y > 1200))))
        |     ( X = 3,
                (((Z = 1,var(Y),Y = 800)
                   | (Z = 1,nonvar(Y),Y =< 800))
               |  ((Z = 2,var(Y),Y = 1000)
                   | (Z = 2,nonvar(Y),Y =< 1000))
               |  ((Z = 3,var(Y),Y = 1200)
                   | (Z = 3,nonvar(Y),Y =< 1200))))).


                     rule(60,S,X,Y,Z) :-
                      xkb_identify(' Group F Division 1 '),


                    (( X = 1,

      ((S = 1,
               (((Z = 1,var(Y),Y = 2400)
                  | (Z = 1,nonvar(Y),Y =< 2400,Y > 800))
               |  ((Z = 2,var(Y),Y = 3000)
                  | (Z = 2,nonvar(Y),Y =< 3000,Y > 1000))
               |  ((Z = 3,var(Y),Y = 3600)
                  | (Z = 3,nonvar(Y),Y =< 3600,Y > 1200))))
        |  (S = 2,
               (((Z = 1,var(Y),Y = 4800)
                  | (Z = 1,nonvar(Y),Y =< 4800,Y > 2400))
               |  ((Z = 2,var(Y),Y = 6000)
                  | (Z = 2,nonvar(Y),Y =< 6000,Y > 3000))
               |  ((Z = 3,var(Y),Y = 7200)
                  | (Z = 3,nonvar(Y),Y =< 7200,Y > 3600))))))

        |     ( X = 2,
     ((S = 1,
               (((Z = 1,var(Y),Y = 1200)
                  | (Z = 1,nonvar(Y),Y =< 1200,Y > 400))
               |  ((Z = 2,var(Y),Y = 1500)
                  | (Z = 2,nonvar(Y),Y =< 1500,Y > 500))
               |  ((Z = 3,var(Y),Y = 1800)
                  | (Z = 3,nonvar(Y),Y =< 1800,Y > 600))))
  | (S = 2,
               (((Z = 1,var(Y),Y = 2400)
                  | (Z = 1,nonvar(Y),Y =< 2400,Y > 1200))
               |  ((Z = 2,var(Y),Y = 3000)
                  | (Z = 2,nonvar(Y),Y =< 3000,Y > 1500))
               |  ((Z = 3,var(Y),Y = 3600)
                  | (Z = 3,nonvar(Y),Y =< 3600,Y > 1800))))))
```

```
        |     ( X = 3,
    ((S = 1,
                (((Z = 1,var(Y),Y = 800)
                  | (Z = 1,nonvar(Y),Y =< 800))
               |  ((Z = 2,var(Y),Y = 1000)
                  | (Z = 2,nonvar(Y),Y =< 1000))
               |  ((Z = 3,var(Y),Y = 1200)
                  | (Z = 3,nonvar(Y),Y =< 1200))))
| (S = 2,
                (((Z = 1,var(Y),Y = 1600)
                  | (Z = 1,nonvar(Y),Y =< 1600,Y > 800))
               |  ((Z = 2,var(Y),Y = 2000)
                  | (Z = 2,nonvar(Y),Y =< 2000,Y > 1000))
               |  ((Z = 3,var(Y),Y = 2400)
                  | (Z = 3,nonvar(Y),Y =< 2400,Y > 1200))))))


        |     ( X = 4,
    ((S = 1,
                (((Z = 1,var(Y),Y = 600)
                  | (Z = 1,nonvar(Y),Y =< 600))
               |  ((Z = 2,var(Y),Y = 750)
                  | (Z = 2,nonvar(Y),Y =< 750))
               |  ((Z = 3,var(Y),Y = 900)
                  | (Z = 3,nonvar(Y),Y =< 900))))
| (S = 2,
                (((Z = 1,var(Y),Y = 1200)
                  | (Z = 1,nonvar(Y),Y =< 1200))
               |  ((Z = 2,var(Y),Y = 1500)
                  | (Z = 2,nonvar(Y),Y =< 1500))
               |  ((Z = 3,var(Y),Y = 1800)
                  | (Z = 3,nonvar(Y),Y =< 1800))))))).


                        rule(61,S,X,Y,Z) :-
                     xkb_identify(' Group F Division 1 '),
                                S = 2,
                   (( X = 1,

               (((Z = 1,var(Y),Y = 6000)
                  | (Z = 1,nonvar(Y),Y =< 6000,Y > 4800))
               |  ((Z = 2,var(Y),Y = 7500)
                  | (Z = 2,nonvar(Y),Y =< 7500,Y > 6000))
               |  ((Z = 3,var(Y),Y = 9000)
                  | (Z = 3,nonvar(Y),Y =< 9000,Y > 7200))))

          |     ( X = 2,
                 (((Z = 1,var(Y),Y = 3000)
                    | (Z = 1,nonvar(Y),Y =< 3000,Y > 2400))
               |  ((Z = 2,var(Y),Y = 3750)
                  | (Z = 2,nonvar(Y),Y =< 3750,Y > 3000))
               |  ((Z = 3,var(Y),Y = 4500)
```

```
          | (Z = 3,nonvar(Y),Y =< 4500,Y > 3600))))

    |     ( X = 3,
           (((Z = 1,var(Y),Y = 2000)
             | (Z = 1,nonvar(Y),Y =< 2000,Y > 1600))
          |  ((Z = 2,var(Y),Y = 2500)
             | (Z = 2,nonvar(Y),Y =< 2500,Y > 2000))
          |  ((Z = 3,var(Y),Y = 3000)
             | (Z = 3,nonvar(Y),Y =< 3000,Y > 2400))))


    |     ( X = 4,
           (((Z = 1,var(Y),Y = 1500)
             | (Z = 1,nonvar(Y),Y =< 1500,Y > 1200))
          |  ((Z = 2,var(Y),Y = 1875)
             | (Z = 2,nonvar(Y),Y =< 1875,Y > 1500))
          |  ((Z = 3,var(Y),Y = 2250)
             | (Z = 3,nonvar(Y),Y =< 2250,Y > 1800))))).


          rule(62,S,X,Y,Z) :-
               xkb_identify(' Group F Division 2 '),

                ((  X = 1,
((S = 1,
         (((Z = 1,var(Y),Y = 1000)
           | (Z = 1,nonvar(Y),Y =< 1000))
        |  ((Z = 2,var(Y),Y = 1250)
           | (Z = 2,nonvar(Y),Y =< 1250))
        |  ((Z = 3,var(Y),Y = 1500)
           | (Z = 3,nonvar(Y),Y =< 1500))))
    | (S = 2,
         (((Z = 1,var(Y),Y = 3000)
           | (Z = 1,nonvar(Y),Y =< 3000))
        |  ((Z = 2,var(Y),Y = 3750)
           | (Z = 2,nonvar(Y),Y =< 3750))
        |  ((Z = 3,var(Y),Y = 4500)
           | (Z = 3,nonvar(Y),Y =< 4500))))))

        |              (  X = 2,
((S = 1,
         (((Z = 1,var(Y),Y = 600)
           | (Z = 1,nonvar(Y),Y =< 600))
        |  ((Z = 2,var(Y),Y = 750)
           | (Z = 2,nonvar(Y),Y =< 750))
        |  ((Z = 3,var(Y),Y = 900)
           | (Z = 3,nonvar(Y),Y =< 900))))
    | (S = 2,
         (((Z = 1,var(Y),Y = 1200)
           | (Z = 1,nonvar(Y),Y =< 1200))
        |  ((Z = 2,var(Y),Y = 1500)
```

```
                    | (Z = 2,nonvar(Y),Y =< 1500))
                |   ((Z = 3,var(Y),Y = 1800)
                    | (Z = 3,nonvar(Y),Y =< 1800))))))).


                      rule(63,S,X,Y,Z) :-
                        xkb_identify(' Group F Division 2 '),


                     (( X = 1,

        ((S = 1,
                ((((Z = 1,var(Y),Y = 3200)
                   | (Z = 1,nonvar(Y),Y =< 3200,Y > 1000))
                | ((Z = 2,var(Y),Y = 4000)
                   | (Z = 2,nonvar(Y),Y =< 4000,Y > 1250))
                | ((Z = 3,var(Y),Y = 4800)
                   | (Z = 3,nonvar(Y),Y =< 4800,Y > 1500))))
         |  (S = 2,
                (((Z = 1,var(Y),Y = 6400)
                   | (Z = 1,nonvar(Y),Y =< 6400,Y > 3000))
                |  ((Z = 2,var(Y),Y = 8000)
                   | (Z = 2,nonvar(Y),Y =< 8000,Y > 3750))
                |  ((Z = 3,var(Y),Y = 9600)
                   | (Z = 3,nonvar(Y),Y =< 9600,Y > 4500))))))

          |     ( X = 2,
      ((S = 1,
                 (((Z = 1,var(Y),Y = 1600)
                   | (Z = 1,nonvar(Y),Y =< 1600,Y > 600))
                |  ((Z = 2,var(Y),Y = 2000)
                   | (Z = 2,nonvar(Y),Y =< 2000,Y > 750))
                |  ((Z = 3,var(Y),Y = 2400)
                   | (Z = 3,nonvar(Y),Y =< 2400,Y > 900))))
  | (S = 2,
                 (((Z = 1,var(Y),Y = 3200)
                   | (Z = 1,nonvar(Y),Y =< 3200,Y > 1200))
                |  ((Z = 2,var(Y),Y = 4000)
                   | (Z = 2,nonvar(Y),Y =< 4000,Y > 1500))
                |  ((Z = 3,var(Y),Y = 4800)
                   | (Z = 3,nonvar(Y),Y =< 4800,Y > 1800))))))

          |     ( X = 3,
      ((S = 1,
                 (((Z = 1,var(Y),Y = 1070)
                   | (Z = 1,nonvar(Y),Y =< 1070))
                |  ((Z = 2,var(Y),Y = 1340)
                   | (Z = 2,nonvar(Y),Y =< 1340))
                |  ((Z = 3,var(Y),Y = 1600)
                   | (Z = 3,nonvar(Y),Y =< 1600))))
  | (S = 2,
                 (((Z = 1,var(Y),Y = 2140)
```

```
                    | (Z = 1,nonvar(Y),Y =< 2140))
                |   ((Z = 2,var(Y),Y = 2680)
                    | (Z = 2,nonvar(Y),Y =< 2680))
                |   ((Z = 3,var(Y),Y = 3200)
                    | (Z = 3,nonvar(Y),Y =< 3200))))))


        |     ( X = 4,
    ((S = 1,
                (((Z = 1,var(Y),Y = 800)
                  | (Z = 1,nonvar(Y),Y =< 800))
                |   ((Z = 2,var(Y),Y = 1000)
                    | (Z = 2,nonvar(Y),Y =< 1000))
                |   ((Z = 3,var(Y),Y = 1200)
                    | (Z = 3,nonvar(Y),Y =< 1200))))
| (S = 2,
                (((Z = 1,var(Y),Y = 1600)
                  | (Z = 1,nonvar(Y),Y =< 1600))
                |   ((Z = 2,var(Y),Y = 2000)
                    | (Z = 2,nonvar(Y),Y =< 2000))
                |   ((Z = 3,var(Y),Y = 2400)
                    | (Z = 3,nonvar(Y),Y =< 2400))))))).



rule(64,S,X,Y,Z) :-
                      xkb_identify(' Group F Division 2 '),
                      X = 1,

      ((S = 1,
                (((Z = 1,var(Y),Y = 1600)
                  | (Z = 1,nonvar(Y), Y =< 1600))
                |   ((Z = 2,var(Y),Y =2000)
                    |(Z = 2,nonvar(Y),Y =< 2000))
                |   ((Z = 3,var(Y),Y = 2400)
                    |(Z = 3,nonvar(Y),Y =< 2400 ))))
         | (S = 2,
                (((Z = 1,var(Y),Y = 3200)
                  | (Z = 1,nonvar(Y),Y =< 3200))
                |   ((Z = 2,var(Y),Y = 4000)
                    |(Z = 2,nonvar(Y),Y =< 4000))
                |   ((Z = 3,var(Y),Y = 4800)
                    |(Z = 3,nonvar(Y),Y =< 4800))))).


                        rule(65,S,X,Y,Z) :-
                        xkb_identify(' Group F Division 2 '),


                     (( X = 1,
```

```
      ((S = 1,
              (((Z = 1,var(Y),Y = 6000)
                 | (Z = 1,nonvar(Y),Y =< 6000,Y > 3200))
               | ((Z = 2,var(Y),Y = 7500)
                 | (Z = 2,nonvar(Y),Y =< 7500,Y > 4000))
               | ((Z = 3,var(Y),Y = 9000)
                 | (Z = 3,nonvar(Y),Y =< 9000,Y > 4800))))
       |  (S = 2,
              (((Z = 1,var(Y),Y = 12000)
                 | (Z = 1,nonvar(Y),Y =< 12000,Y > 6400))
               | ((Z = 2,var(Y),Y = 15000)
                 | (Z = 2,nonvar(Y),Y =< 15000,Y > 8000))
               | ((Z = 3,var(Y),Y = 18000)
                 | (Z = 3,nonvar(Y),Y =< 18000,Y >
                                                 9600))))))

     |    ( X = 2,
    ((S = 1,
              (((Z = 1,var(Y),Y = 3000)
                 | (Z = 1,nonvar(Y),Y =< 3000,Y > 1600))
               | ((Z = 2,var(Y),Y = 3750)
                 | (Z = 2,nonvar(Y),Y =< 3750,Y > 2000))
               | ((Z = 3,var(Y),Y = 4500)
                 | (Z = 3,nonvar(Y),Y =< 4500,Y > 2400))))
| (S = 2,
              (((Z = 1,var(Y),Y = 6000)
                 | (Z = 1,nonvar(Y),Y =< 6000,Y > 3200))
               | ((Z = 2,var(Y),Y = 7500)
                 | (Z = 2,nonvar(Y),Y =< 7500,Y > 4000))
               | ((Z = 3,var(Y),Y = 9000)
                 | (Z = 3,nonvar(Y),Y =< 9000,Y > 4800))))))

     |    ( X = 3,
    ((S = 1,
              (((Z = 1,var(Y),Y = 2000)
                 | (Z = 1,nonvar(Y),Y =< 2000,Y > 1070))
               | ((Z = 2,var(Y),Y = 2500)
                 | (Z = 2,nonvar(Y),Y =< 2500,Y > 1340))
               | ((Z = 3,var(Y),Y = 3000)
                 | (Z = 3,nonvar(Y),Y =< 3000,Y > 1600))))
| (S = 2,
              (((Z = 1,var(Y),Y = 4000)
                 | (Z = 1,nonvar(Y),Y =< 4000,Y > 2140))
               | ((Z = 2,var(Y),Y = 5000)
                 | (Z = 2,nonvar(Y),Y =< 5000,Y > 2680))
               | ((Z = 3,var(Y),Y = 6000)
                 | (Z = 3,nonvar(Y),Y =< 6000,Y > 3200))))))


     |    ( X = 4,
    ((S = 1,
              (((Z = 1,var(Y),Y = 1500)
```

```
                    | (Z = 1,nonvar(Y),Y =< 1500,Y > 800))
                |   ((Z = 2,var(Y),Y = 1875)
                    | (Z = 2,nonvar(Y),Y =< 1875,Y > 1000))
                |   ((Z = 3,var(Y),Y = 2250)
                    | (Z = 3,nonvar(Y),Y =< 2250,Y > 1200))))
| (S = 2,
                  (((Z = 1,var(Y),Y = 3000)
                    | (Z = 1,nonvar(Y),Y =< 3000,Y > 1600))
                |   ((Z = 2,var(Y),Y = 3750)
                    | (Z = 2,nonvar(Y),Y =< 3750,Y > 2000))
                |   ((Z = 3,var(Y),Y = 4500)
                    | (Z = 3,nonvar(Y),Y =< 4500,Y >
                                          2400))))))).


        rule(66,S,X,Y,Z) :-
                    xkb_identify(' Group F Division 2 '),

                     (( X = 1,

      ((S = 1,
              (((Z = 1,var(Y),Y = 9000)
                  | (Z = 1,nonvar(Y),Y =< 9000,Y > 6000))
                | ((Z = 2,var(Y),Y = 11250)
                  | (Z = 2,nonvar(Y),Y =< 11250,Y > 7500))
                | ((Z = 3,var(Y),Y = 13500)
                  | (Z = 3,nonvar(Y),Y = 13500,Y > 9000))))
       |   (S = 2,
              (((Z = 1,var(Y),Y = 18000)
                  | (Z = 1,nonvar(Y),Y =< 18000,Y > 12000))
              |   ((Z = 2,var(Y),Y = 22500)
                  | (Z = 2,nonvar(Y),Y =< 22500,Y > 15000))
              |   ((Z = 3,var(Y),Y = 27000)
                  | (Z = 3,nonvar(Y),Y =< 27000,Y >
                                        18000))))))

          |     ( X = 2,
     ((S = 1,
              (((Z = 1,var(Y),Y = 4500)
                  | (Z = 1,nonvar(Y),Y =< 4500,Y > 3000))
              |   ((Z = 2,var(Y),Y = 5625)
                  | (Z = 2,nonvar(Y),Y =< 5625,Y > 3750))
              |   ((Z = 3,var(Y),Y = 6750)
                  | (Z = 3,nonvar(Y),Y =< 6750,Y > 4500))))
| (S = 2,
              (((Z = 1,var(Y),Y = 9000)
                  | (Z = 1,nonvar(Y),Y =< 9000,Y > 6000))
              |   ((Z = 2,var(Y),Y = 11250)
                  | (Z = 2,nonvar(Y),Y =< 11250,Y > 7500))
              |   ((Z = 3,var(Y),Y = 13500)
                  | (Z = 3,nonvar(Y),Y =< 13500,Y >
                                        9000),,))))
```

```
        |     ( X = 3,
    ((S = 1,
                (((Z = 1,var(Y),Y = 3000)
                  | (Z = 1,nonvar(Y),Y =< 3000,Y > 2000))
              |  ((Z = 2,var(Y),Y = 3750)
                  | (Z = 2,nonvar(Y),Y =< 3750,Y > 2500))
              |  ((Z = 3,var(Y),Y = 4500)
                  | (Z = 3,nonvar(Y),Y =< 4500,Y > 3000))))
| (S = 2,
                (((Z = 1,var(Y),Y = 6000)
                  | (Z = 1,nonvar(Y),Y =< 6000,Y > 4000))
              |  ((Z = 2,var(Y),Y = 7500)
                  | (Z = 2,nonvar(Y),Y =< 7500,Y > 5000))
              |  ((Z = 3,var(Y),Y = 9000)
                  | (Z = 3,nonvar(Y),Y =< 9000,Y > 6000))))))

        |     ( X = 4,
     ((S = 1,
              (((Z = 1,var(Y),Y = 2250)
                 | (Z = 1,nonvar(Y),Y =< 2250,Y =< 1500))
               | ((Z = 2,var(Y),Y = 2812)
                 | (Z = 2,nonvar(Y),Y =< 2812,Y =< 1875))
               | ((Z = 3,var(Y),Y = 3375)
                 | (Z = 3,nonvar(Y),Y = 3375,Y =< 2250))))
      |  (S = 2,
              (((Z = 1,var(Y),Y = 4500)
                 | (Z = 1,nonvar(Y),Y =< 4500,Y =< 3000))
              |  ((Z = 2,var(Y),Y = 5624)
                  | (Z = 2,nonvar(Y),Y =< 5624,Y =< 3750))
              |  ((Z = 3,var(Y),Y = 6750)
                  | (Z = 3,nonvar(Y),Y =< 6750,Y =<
                                                 4500))))))

        |     ( X = 5,
    ((S = 1,
              (((Z = 1,var(Y),Y = 1800)
                 | (Z = 1,nonvar(Y),Y =< 1800))
              |  ((Z = 2,var(Y),Y = 2250)
                  | (Z = 2,nonvar(Y),Y =< 2250))
              |  ((Z = 3,var(Y),Y = 2700)
                  | (Z = 3,nonvar(Y),Y =< 2700))))
| (S = 2,
              (((Z = 1,var(Y),Y = 3600)
                 | (Z = 1,nonvar(Y),Y =< 3600))
              |  ((Z = 2,var(Y),Y = 4500)
                  | (Z = 2,nonvar(Y),Y =< 4500))
              |  ((Z = 3,var(Y),Y = 5400)
                  | (Z = 3,nonvar(Y),Y =< 5400))))))
        |     ( X = 6,
    ((S = 1,
              (((Z = 1,var(Y),Y = 1500)
                 | (Z = 1,nonvar(Y),Y =< 1500))
```

```
                |   ((Z = 2,var(Y),Y = 1875)
                    | (Z = 2,nonvar(Y),Y =< 1875))
                |   ((Z = 3,var(Y),Y = 2250)
                    | (Z = 3,nonvar(Y),Y =< 2250))))
| (S = 2,
                  (((Z = 1,var(Y),Y = 3000)
                    | (Z = 1,nonvar(Y),Y =< 3000))
                |   ((Z = 2,var(Y),Y = 3750)
                    | (Z = 2,nonvar(Y),Y =< 3750))
                |   ((Z = 3,var(Y),Y = 4500)
                    | (Z = 3,nonvar(Y),Y =< 4500))))))).


                        rule(67,S,X,Y,Z) :-
                        xkb_identify(' Group F Division 2 '),

                ((  X = 1,
    ((S = 1,
                    (((Z = 1,var(Y),Y = 'unlimited')
                      | (Z = 1,nonvar(Y),Y > 9000))
                |   ((Z = 2,var(Y),Y = 'unlimited')
                      | (Z = 2,nonvar(Y),Y > 11250))
                |   ((Z = 3,var(Y),Y = 'unlimited')
                      | (Z = 3,nonvar(Y),Y > 13500))))
| (S = 2,
                    (((Z = 1,var(Y),Y = 'unlimited')
                      | (Z = 1,nonvar(Y),Y > 18000))
                |   ((Z = 2,var(Y),Y = 'unlimited')
                      | (Z = 2,nonvar(Y),Y > 22500))
                |   ((Z = 3,var(Y),Y = 'unlimited')
                      | (Z = 3,nonvar(Y),Y > 27000))))))


        |     (  X = 2,
    ((S = 1,
                    (((Z = 1,var(Y),Y = 'unlimited')
                      | (Z = 1,nonvar(Y),Y > 4500))
                |   ((Z = 2,var(Y),Y = 'unlimited')
                      | (Z = 2,nonvar(Y),Y > 5625))
                |   ((Z = 3,var(Y),Y = 'unlimited')
                      | (Z = 3,nonvar(Y),Y > 6750))))
| (S = 2,
                    (((Z = 1,var(Y),Y = 'unlimited')
                      | (Z = 1,nonvar(Y),Y > 9000))
                |   ((Z = 2,var(Y),Y = 'unlimited')
                      | (Z = 2,nonvar(Y),Y > 11250))
                |   ((Z = 3,var(Y),Y = 'unlimited')
                      | (Z = 3,nonvar(Y),Y > 13500))))))


        |     (  X = 3,
    ((S = 1,
```

```
                (((Z = 1,var(Y),Y = 'unlimited')
                  | (Z = 1,nonvar(Y),Y > 3000))
              |  ((Z = 2,var(Y),Y = 'unlimited')
                  | (Z = 2,nonvar(Y),Y > 3750))
              |  ((Z = 3,var(Y),Y = 'unlimited')
                  | (Z = 3,nonvar(Y),Y > 4500))))
| (S = 2,
                (((Z = 1,var(Y),Y = 'unlimited')
                  | (Z = 1,nonvar(Y),Y > 6000))
              |  ((Z = 2,var(Y),Y = 'unlimited')
                  | (Z = 2,nonvar(Y),Y > 7500))
              |  ((Z = 3,var(Y),Y = 'unlimited')
                  | (Z = 3,nonvar(Y),Y > 9000))))))


        |      ( X = 4,
     ((S = 1,
              (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 2250))
               | ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 2812))
               | ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 3375))))
      |  (S = 2,
            (((Z = 1,var(Y),Y = 'unlimited')
              | (Z = 1,nonvar(Y),Y > 4500))
            |  ((Z = 2,var(Y),Y = 'unlimited')
              | (Z = 2,nonvar(Y),Y > 5624))
            |  ((Z = 3,var(Y),Y = 'unlimited')
              | (Z = 3,nonvar(Y),Y > 6750))))))

        |     ( X = 5,
    ((S = 1,
              (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 1800))
            |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 2250))
            |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 2700))))
| (S = 2,
              (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 3600))
            |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 4500))
            |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 5400))))))
        |     ( X = 6,
    ((S = 1,
              (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 1500))
            |  ((Z = 2,var(Y),Y = 'unlimited')
                | (Z = 2,nonvar(Y),Y > 1875))
```

```
            |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 2250))))
| (S = 2,
            (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 3000))
            |  ((Z = 2,var(Y),Y = 'unlimited'
                | (Z = 2,nonvar(Y),Y > 3750))
            |  ((Z = 3,var(Y),Y = 'unlimited')
                | (Z = 3,nonvar(Y),Y > 4500))))))
      |     ( nonvar(X),X > 6)).


          rule(68,S,X,Y,Z) :-
                xkb_identify(' Group F Division 3 '),

              (( X = 1,
((S = 1,
        (((Z = 1,var(Y),Y = 1600)
            | (Z = 1,nonvar(Y),Y =< 1600))
        | ((Z = 2,var(Y),Y = 2000)
            | (Z = 2,nonvar(Y),Y =< 2000))
        | ((Z = 3,var(Y),Y = 2400)
            | (Z = 3,nonvar(Y),Y =< 2400))))
  | (S = 2,
        (((Z = 1,var(Y),Y = 4800)
            | (Z = 1,nonvar(Y),Y =< 4800))
        |  ((Z = 2,var(Y),Y = 6000)
            | (Z = 2,nonvar(Y),Y =< 6000))
        |  ((Z = 3,var(Y),Y = 7200)
            | (Z = 3,nonvar(Y),Y =< 7200))))))

        |           (  X = 2,
((S = 1,
        (((Z = 1,var(Y),Y = 800)
            | (Z = 1,nonvar(Y),Y =< 800))
        |  ((Z = 2,var(Y),Y = 1000)
            | (Z = 2,nonvar(Y),Y =< 1000))
        |  ((Z = 3,var(Y),Y = 1200)
            | (Z = 3,nonvar(Y),Y =< 1200))))
  | (S = 2,
        (((Z = 1,var(Y),Y = 1600)
            | (Z = 1,nonvar(Y),Y =< 1600))
        |  ((Z = 2,var(Y),Y = 2000)
            | (Z = 2,nonvar(Y),Y =< 2000))
        |  ((Z = 3,var(Y),Y = 2400)
            | (Z = 3,nonvar(Y),Y =< 2400)))))))).


            rule(69,S,X,Y,Z) :-
            xkb_identify(' Group F Division 3 '),
```

```
                    (( X = 1,

      ((S = 1,
              (((Z = 1,var(Y),Y = 4800)
                 | (Z = 1,nonvar(Y),Y =< 4800,Y > 1600))
               | ((Z = 2,var(Y),Y = 6000)
                  | (Z = 2,nonvar(Y),Y =< 6000,Y > 2000))
               | ((Z = 3,var(Y),Y = 7200)
                  | (Z = 3,nonvar(Y),Y =< 7200,Y > 2400))))
       |  (S = 2,
             (((Z = 1,var(Y),Y = 9600)
                | (Z = 1,nonvar(Y),Y =< 9600,Y > 4800))
             |  ((Z = 2,var(Y),Y = 12000)
                 | (Z = 2,nonvar(Y),Y =< 12000,Y > 6000))
             |  ((Z = 3,var(Y),Y = 14400)
                 | (Z = 3,nonvar(Y),Y =< 14400,Y >
                                              7200))))))

         |     ( X = 2,
    ((S = 1,
              (((Z = 1,var(Y),Y = 2400)
                 | (Z = 1,nonvar(Y),Y =< 2400,Y > 800))
             |  ((Z = 2,var(Y),Y = 3000)
                 | (Z = 2,nonvar(Y),Y =< 3000,Y > 1000))
             |  ((Z = 3,var(Y),Y = 3600)
                 | (Z = 3,nonvar(Y),Y =< 3600,Y > 1200))))
| (S = 2,
              (((Z = 1,var(Y),Y = 4800)
                 | (Z = 1,nonvar(Y),Y =< 4800,Y > 1600))
             |  ((Z = 2,var(Y),Y = 6000)
                 | (Z = 2,nonvar(Y),Y =< 6000,Y > 2000))
             |  ((Z = 3,var(Y),Y = 7200)
                 | (Z = 3,nonvar(Y),Y =< 7200,Y > 2400))))))

         |     ( X = 3,
    ((S = 1,
              (((Z = 1,var(Y),Y = 1600)
                 | (Z = 1,nonvar(Y),Y =< 1600))
             |  ((Z = 2,var(Y),Y = 2000)
                 | (Z = 2,nonvar(Y),Y =< 2000))
             |  ((Z = 3,var(Y),Y = 2400)
                 | (Z = 3,nonvar(Y),Y =< 2400))))
| (S = 2,
              (((Z = 1,var(Y),Y = 3200)
                 | (Z = 1,nonvar(Y),Y =< 3200))
             |  ((Z = 2,var(Y),Y = 2000)
                 | (Z = 2,nonvar(Y),Y =< 2000))
             |  ((Z = 3,var(Y),Y = 2400)
                 | (Z = 3,nonvar(Y),Y =< 2400))))))


         |     ( X = 4,
```

```
    ((S = 1,
              (((Z = 1,var(Y),Y = 1200)
                | (Z = 1,nonvar(Y),Y =< 1200))
             |  ((Z = 2,var(Y),Y = 1500)
                | (Z = 2,nonvar(Y),Y =< 1500))
             |  ((Z = 3,var(Y),Y = 1800)
                | (Z = 3,nonvar(Y),Y =< 1800))))
| (S = 2,
              (((Z = 1,var(Y),Y = 2400)
                | (Z = 1,nonvar(Y),Y =< 2400))
             |  ((Z = 2,var(Y),Y = 3000)
                | (Z = 2,nonvar(Y),Y =< 3000))
             |  ((Z = 3,var(Y),Y = 3600)
                | (Z = 3,nonvar(Y),Y =< 3600))))))).



rule(70,S,X,Y,Z) :-
                   xkb_identify(' Group F Division 3 '),
                   X = 1,

     ((S = 1,
              (((Z = 1,var(Y),Y = 1200)
               | (Z = 1,nonvar(Y), Y =< 1200))
             |  ((Z = 2,var(Y),Y =1500)
                |(Z = 2,nonvar(Y),Y =< 1500))
             |  ((Z = 3,var(Y),Y = 1800)
                |(Z = 3,nonvar(Y),Y =< 1800 ))))
        | (S = 2,
              (((Z = 1,var(Y),Y = 2400)
               | (Z = 1,nonvar(Y),Y =< 2400))
             |  ((Z = 2,var(Y),Y = 3000)
                |(Z = 2,nonvar(Y),Y =< 3000))
             |  ((Z = 3,var(Y),Y = 3600)
                |(Z = 3,nonvar(Y),Y =< 3600))))).


                      rule(71,S,X,Y,Z) :-
                   xkb_identify(' Group F Division 3 '),
                                ( X = 1,
     ((S = 1,
              (((Z = 1,var(Y),Y = 5600)
                | (Z = 1,nonvar(Y),Y =< 5600,Y > 4800))
             |  ((Z = 2,var(Y),Y = 7000)
                | (Z = 2,nonvar(Y),Y =< 7000,Y > 6000))
             |  ((Z = 3,var(Y),Y = 8400)
                | (Z = 3,nonvar(Y),Y =< 8400,Y > 7200))))
         |  (S = 2,
              (((Z = 1,var(Y),Y = 11200)
                | (Z = 1,nonvar(Y),Y =< 11200,Y > 9600))
             |  ((Z = 2,var(Y),Y = 14000)
```

```
                | (Z = 2,nonvar(Y),Y =< 14000,Y > 12000))
            |  ((Z = 3,var(Y),Y = 16800)
                | (Z = 3,nonvar(Y),Y =< 16800,Y >
                                              14400))))))).


        rule(72,S,X,Y,Z) :-
                xkb_identify(' Group F Division 3 '),
                    X = 1,
                    Y = 'unlimited',
                    prop(low_fire_load).


            rule(73,S,X,Y,Z) :-
                xkb_identify(' Group F Division 3 '),
                        prop(storage_garage),
                        ((var(X),X = 'less than 7')
                          | (nonvar(X),X =< 6)),
                        ((var(Y),Y = 10000)
                         | (nonvar(Y),Y =< 10000)).




    rule(74,S,X,Y,Z) :-
              xkb_identify(' Group F Division 3 '),

               (( X = 1,

 ((S = 1,
        (((Z = 1,var(Y),Y = 'unlimited')
           | (Z = 1,nonvar(Y),Y > 4800))
        | ((Z = 2,var(Y),Y = 'unlimited')
           | (Z = 2,nonvar(Y),Y > 6000))
        | ((Z = 3,var(Y),Y = 'unlimited')
           | (Z = 3,nonvar(Y),Y > 7200))))
  |  (S = 2,
        (((Z = 1,var(Y),Y = 'unlimited')
           | (Z = 1,nonvar(Y),Y > 9600))
        |  ((Z = 2,var(Y),Y = 'unlimited')
           | (Z = 2,nonvar(Y),Y > 12000))
        |  ((Z = 3,var(Y),Y = 'unlimited')
           | (Z = 3,nonvar(Y),Y > 14400))))))

   |     ( X = 2,
((S = 1,
         (((Z = 1,var(Y),Y = 7200)
           | (Z = 1,nonvar(Y),Y =< 7200,Y > 2400))
        |  ((Z = 2,var(Y),Y = 9000)
           | (Z = 2,nonvar(Y),Y =< 9000,Y > 3000))
```

```
                | ((Z = 3,var(Y),Y = 10800)
                  | (Z = 3,nonvar(Y),Y =< 10800,Y > 3600))))
| (S = 2,
              (((Z = 1,var(Y),Y = 14400)
                | (Z = 1,nonvar(Y),Y =< 14400,Y > 4800))
              | ((Z = 2,var(Y),Y = 18000)
                | (Z = 2,nonvar(Y),Y =< 18000,Y > 6000))
              | ((Z = 3,var(Y),Y = 21600)
                | (Z = 3,nonvar(Y),Y =< 21600,Y >
                                                7200))))))
        |     ( X = 3,
    ((S = 1,
              (((Z = 1,var(Y),Y = 4800)
                | (Z = 1,nonvar(Y),Y =< 4800,Y > 1600))
              | ((Z = 2,var(Y),Y = 6000)
                | (Z = 2,nonvar(Y),Y =< 6000,Y > 2000))
              | ((Z = 3,var(Y),Y = 7200)
                | (Z = 3,nonvar(Y),Y =< 7200,Y > 2400))))
| (S = 2,
              (((Z = 1,var(Y),Y = 9600)
                | (Z = 1,nonvar(Y),Y =< 9600,Y > 3200))
              | ((Z = 2,var(Y),Y = 12000)
                | (Z = 2,nonvar(Y),Y =< 12000,Y > 4000))
              | ((Z = 3,var(Y),Y = 14400)
                | (Z = 3,nonvar(Y),Y =< 14400,Y >
                                                4800))))))

        |     ( X = 4,
     ((S = 1,
              (((Z = 1,var(Y),Y = 3600)
                | (Z = 1,nonvar(Y),Y =< 3600,Y > 1200))
              | ((Z = 2,var(Y),Y = 4500)
                | (Z = 2,nonvar(Y),Y =< 4500,Y > 1500))
              | ((Z = 3,var(Y),Y = 5400)
                | (Z = 3,nonvar(Y),Y =< 5400,Y > 1800))))
      | (S = 2,
              (((Z = 1,var(Y),Y = 7200)
                | (Z = 1,nonvar(Y),Y =< 7200,Y > 2400))
              | ((Z = 2,var(Y),Y = 9000)
                | (Z = 2,nonvar(Y),Y =< 9000,Y > 3000))
              | ((Z = 3,var(Y),Y = 10800)
                | (Z = 3,nonvar(Y),Y =< 10800,Y >
                                                3600))))))

        |     ( X = 5,
    ((S = 1,
              (((Z = 1,var(Y),Y = 2880)
                | (Z = 1,nonvar(Y),Y =< 2880))
              | ((Z = 2,var(Y),Y = 3600)
                | (Z = 2,nonvar(Y),Y =< 3600))
              | ((Z = 3,var(Y),Y = 4320)
                | (Z = 3,nonvar(Y),Y =< 4320))))
```

```
| (S = 2,
                (((Z = 1,var(Y),Y = 5760)
                   | (Z = 1,nonvar(Y),Y =< 5760))
               |  ((Z = 2,var(Y),Y = 7200)
                   | (Z = 2,nonvar(Y),Y =< 7200))
               |  ((Z = 3,var(Y),Y = 8640)
                   | (Z = 3,nonvar(Y),Y =< 8640))))))
        |     ( X = 6,
    ((S = 1,
                (((Z = 1,var(Y),Y = 2400)
                   | (Z = 1,nonvar(Y),Y =< 2400))
               |  ((Z = 2,var(Y),Y = 3000)
                   | (Z = 2,nonvar(Y),Y =< 3000))
               |  ((Z = 3,var(Y),Y = 3600)
                   | (Z = 3,nonvar(Y),Y =< 3600))))
| (S = 2,
                (((Z = 1,var(Y),Y = 4800)
                   | (Z = 1,nonvar(Y),Y =< 4800))
               |  ((Z = 2,var(Y),Y = 6000)
                   | (Z = 2,nonvar(Y),Y =< 6000))
               |  ((Z = 3,var(Y),Y = 7200)
                   | (Z = 3,nonvar(Y),Y =< 7200))))))).

                    rule(75,S,X,Y,Z) :-
                      xkb_identify(' Group F Division 3 '),


          (( X = 2,

    ((S = 1,
                (((Z = 1,var(Y),Y = 'unlimited')
                   | (Z = 1,nonvar(Y),Y > 7200))
               |  ((Z = 2,var(Y),Y = 'unlimited')
                   | (Z = 2,nonvar(Y),Y > 9000))
               |  ((Z = 3,var(Y),Y = 'unlimited')
                   | (Z = 3,nonvar(Y),Y > 10800))))
| (S = 2,
                (((Z = 1,var(Y),Y = 'unlimited')
                   | (Z = 1,nonvar(Y),Y > 14400))
               |  ((Z = 2,var(Y),Y = 'unlimited')
                   | (Z = 2,nonvar(Y),Y > 18000))
               |  ((Z = 3,var(Y),Y = 'unlimited')
                   | (Z = 3,nonvar(Y),Y > 21600))))))

        |     ( X = 3,
    ((S = 1,
                (((Z = 1,var(Y),Y = 'unlimited')
                   | (Z = 1,nonvar(Y),Y > 4800))
               |  ((Z = 2,var(Y),Y = 'unlimited')
                   | (Z = 2,nonvar(Y),Y > 6000))
               |  ((Z = 3,var(Y),Y = 'unlimited')
                   | (Z = 3,nonvar(Y),Y > 7200))))
```

```
| (S = 2,
                (((Z = 1,var(Y),Y = 'unlimited')
                  | (Z = 1,nonvar(Y),Y > 9600))
              |  ((Z = 2,var(Y),Y = 'unlimited')
                  | (Z = 2,nonvar(Y),Y > 12000))
              |  ((Z = 3,var(Y),Y = 'unlimited')
                  | (Z = 3,nonvar(Y),Y > 14400))))))
          |      ( X = 4,
      ((S = 1,
              (((Z = 1,var(Y),Y = 'unlimited')
                 | (Z = 1,nonvar(Y),Y > 3600))
              | ((Z = 2,var(Y),Y = 'unlimited')
                 | (Z = 2,nonvar(Y),Y > 4500))
              | ((Z = 3,var(Y),Y = 'unlimited')
                 | (Z = 3,nonvar(Y),Y > 5400))))
       |  (S = 2,
             (((Z = 1,var(Y),Y = 'unlimited')
                | (Z = 1,nonvar(Y),Y > 7200))
             |  ((Z = 2,var(Y),Y = 'unlimited')
                 | (Z = 2,nonvar(Y),Y > 9000))
             |  ((Z = 3,var(Y),Y = 'unlimited')
                 | (Z = 3,nonvar(Y),Y > 10800))))))

        |     ( X = 5,
    ((S = 1,
                (((Z = 1,var(Y),Y = 'unlimited')
                  | (Z = 1,nonvar(Y),Y > 2880))
              |  ((Z = 2,var(Y),Y = 'unlimited')
                  | (Z = 2,nonvar(Y),Y > 3600))
              |  ((Z = 3,var(Y),Y = 'unlimited')
                  | (Z = 3,nonvar(Y),Y > 4320))))
| (S = 2,
                (((Z = 1,var(Y),Y = 'unlimited')
                   | (Z = 1,nonvar(Y),Y > 5760))
              |  ((Z = 2,var(Y),Y = 'unlimited')
                  | (Z = 2,nonvar(Y),Y > 7200))
              |  ((Z = 3,var(Y),Y = 'unlimited')
                  | (Z = 3,nonvar(Y),Y > 8640))))))
        |     ( X = 6,
    ((S = 1,
                (((Z = 1,var(Y),Y = 'unlimited')
                  | (Z = 1,nonvar(Y),Y > 2400))
              |  ((Z = 2,var(Y),Y = 'unlimited')
                  | (Z = 2,nonvar(Y),Y > 3000))
              |  ((Z = 3,var(Y),Y = 'unlimited')
                  | (Z = 3,nonvar(Y),Y > 3600))))
| (S = 2,
                (((Z = 1,var(Y),Y = 'unlimited')
                  | (Z = 1,nonvar(Y),Y > 4800))
              |  ((Z = 2,var(Y),Y = 'unlimited')
                  | (Z = 2,nonvar(Y),Y > 6000))
              |  ((Z = 3,var(Y),Y = 'unlimited')
```

```
                    | (Z = 3,nonvar(Y),Y > 7200))))))
          |     ( nonvar(X),X > 6)).



/*
 *  xkb_question(Property_or_Parameter,Question)
 *    Each of these clauses provides a question or a simple
 *    menu to be used by the utility predicates in the
 *    XSHELL.PL file to ask the user whether the building to
      be identified has the property or what value the
      parameter takes for the building.
 */

xkb_question(assembly_occupancy,
          [ ' Is the building used for assembly
              occupancies?',
            'e.g, theatre,art gallery, church, club, court
             room, dance hall, day-care centre, gymnasium,
             lecture hall, library, licensed beverage
             establishment, lodge room, museum, passenger
             station, depot, recreational pier',
         '   restaurant, school, nonresidential undertaking
               premise']).

xkb_question(performing_arts,
          [ ' Is the building used for the production and
              viewing of the performing arts?']).

xkb_question(arena_type,
          [ ' Is the building used as an arena ?',
            'e.g,ice rink, indoor swimming pool']).

xkb_question(open_air,
         [ ' Is the building used for the congregation or
             gathering of persons participating in or
             viewing open air activities?',
       '     e.g, amusement park, bleach, grandstand,
              stadium.']).

xkb_question(institutional_occupancy,
       [ ' Is the building used as an institution?',
   '         e.g, jail, penitentiary, police station with
             detention quarters', prison, hospital,
              reformatory, convalescent home, home for the
              aged, nursing home, orphanage,sanitorium.']).


xkb_question(penal_detention,
      [ ' Is the building used to detain people for penal or
```

```
',        'correctional purposes,or for involuntary
           detention?',
         ' e.g, jail, penitentiary, police station with
           detention, quarters, prison, psychiatric hospital
           with detention quarters, reformatory with detention
           quarters.']).

xkb_question(residential_occupancy,
     [ 'Is the building used for sleeping accommodations',
       'excluding institutions ?']).

xkb_question(business_occupancy,
     [ ' Is the building used for conducting business and',
       ' rendering of professional and personal
         services?']).

xkb_question(mercantile_occupancy,
     [ ' Is the building used for displaying,or selling of
         retail, goods, wares or merchandise?',
     ' e.g,department store, market, shop, supermarket']).

xkb_question(industrial_occupancy,
     [ ' Is the building used for industrial occupancy?',
       ' i.e, assembling, fabricating, manufacturing,
         processing, repairing or storing of goods and
         materials.']).


xkb_question(high_combustible_content,
     [ 'Does the building contain highly combustible and',
      ' flammable or explosive materials ?',
      ' e.g,bulk plant for flammable liquid,cereal mill,',
     ' chemical or processing plant,distillery,flour mill',
    '  grain elevator, paint factory, waste paper processing
      plant.']).


xkb_question(medium_combustible_content,
      [ ' Does the building have combustible content more
          than 50 kg/m2 or 1200 MJ/m2 ?',
       ' e.g, aircraft hangar, box factory,candy plant,',
       'cold storage plant, dry cleaning plant, electrical
        substation, factory, freight depot, laboratory,
        laundry except self-service, mattress factory,
        planing mill, printing plant, repair garage,
        salesroom, service station, tire storage,',
      ' warehouse, woodworking factory, workshop.']).

xkb_question(storage_garage,
         [' Is the building used as a storage garage ?']).

xkb_question(low_fire_load,
```

```
        [' Is the buildung used solely for low fire load
            occupancies such as',
          ' (i)  power generating plants, or',
          ' (ii) plants for the manufacture or storage of
                 noncombustible materials such as asbestos,
                 brick, cement, concrete or steel']).


xkb_question(number_storeys,
      [' What is the building height ? ',
      ' (a) 1 storey ',
      ' (b) 2 storeys ',
      ' (c) 3 storeys ',
      ' (d) 4 storeys ',
      ' (e) 5 storeys ',
      ' (f) 6 storeys ',
      ' (g) greater  than 6 storeys  ']).

xkb_question(auditorium_floor_height,
      [' Is the auditorium floor less than 5 metres ',
       ' above or below grade ? ']).


xkb_question(occupant_load_floor,
      [ ' What is the occupant load of the auditorium
          floor?',
        ' (a)   less than 300 persons. ',
        ' (b)   300 to 600 persons. ']).

xkb_question(building_area,
      [' What is the building area ?',
      ' (1)  less than 600 m2.',
      ' (2)  greater than 600 m2. ']).


xkb_question(number_streets_building_faces,
        [' How many streets the building faces ?',
         ' (a) 1 street   (b) 2 streets  (c)  3 streets ']).


xkb_question(unsprinklered,
        [' Is the building unsprinklered ?']).

xkb_question(area_unsprinklered_Group_A,
       [' What is the building area ?',
        ' choose the closest number greater than the building
          area',
        '',
       '(a) 400  m2  (b) 500  m2  (c) 600  m2  (d) 800  m2
        (e) 1000 m2 ',
       '(f) 1200 m2  (g) 1250 m2  (h) 1500 m2  (j) 1600 m2
        (k) 1800 m2 ',
```

```
      '(l) 2000 m2  (m) 2400 m2  (n) 2500 m2  (p) 3000 m2
       (q) 3600 m2 ',
      '(r) 4000 m2  (s) 5000 m2  (t) 6000 m2  (u) greater
          than 6000 m2']).


 xkb_question(area_sprinklered_Group_A,
       [' What is the building area ?',
        ' choose the closest number greater than the building
          area',
        '',
      '(a) 800  m2  (b) 1000 m2  (c) 1200 m2  (d) 1600 m2
       (e) 2000 m2 ',
      '(f) 2400 m2  (g) 2500 m2  (h) 3000 m2  (j) 3200 m2
       (k) 3600 m2 ',
      '(l) 4000 m2  (m) 4800 m2  (n) 5000 m2  (p) 6000 m2
       (q) 7200 m2 ',
      '(r) 8000 m2  (s) 10000 m2 (t) 12000 m2 (u) greater
          than 12000 m2']).

xkb_question(area_unsprinklered_Group_B,
       [' What is the building area ?',
        ' choose the closest number greater than the building
          area',
        '',
      '(a) 250  m2  (b) 500  m2  (c) 1000  m2  (d) greater
          than 1000  m2 ']).


xkb_question(area_sprinklered_Group_B,
       [' What is the building area ?',
        ' choose the closest number greater than the building
          area',
        '',
      '(a) 500  m2  (b) 1600  m2  (c) 2400  m2  (d) 8000  m2 ',
      '(e) 12000 m2 (f) unlimited ']).

xkb_question(area_unsprinklered_Group_C,
       [' What is the building area ?',
        ' choose the closest number greater than the building
          area',
        '',
      '(a) 600  m2  (b) 750  m2  (c) 800  m2  (d) 900  m2
       (e) 1000 m2 ',
      '(f) 1125 m2  (g) 1200 m2  (h) 1350 m2  (j) 1500 m2
       (k) 1800 m2 ',
      '(l) 2000 m2  (m) 2250 m2  (n) 2400 m2  (p) 2500 m2
       (q) 2700 m2 ',
      '(r) 3000 m2  (s) 3600 m2  (t) 3750 m2  (u) 4000 m2
       (v) 4500 m2',
      '(w) 5000 m2  (x) 6000 m2  (y) unlimited ']).
```

```
xkb_question(area_sprinklered_Group_C,
     [' What is the building area ?',
      ' choose the closest number greater than the building
        area',
      '',
     '(a) 1200  m2  (b) 1500 m2  (c) 1600 m2  (d) 1800 m2
      (e) 2000 m2 ',
     '(f) 2250 m2   (g) 2400 m2  (h) 2700 m2  (j) 3000 m2
      (k) 3600 m2 ',
     '(l) 4000 m2   (m) 4500 m2  (n) 4800 m2  (p) 5000 m2
      (q) 5400 m2 ',
     '(r) 6000 m2   (s) 7200 m2  (t) 7500 m2  (u) 8000 m2
      (v) 9000 m2',
     '(w) 10000 m2  (x) 12000 m2 (y) unlimited ']).


xkb_question(area_unsprinklered_Group_D,
     [' What is the building area ?',
      ' choose the closest number greater than the building
        area',
      '',
     '(a) 800  m2  (b) 1000 m2  (c) 1200 m2  (d) 1250 m2
      (e) 1500 m2 ',
     '(f) 1600 m2  (g) 2000 m2  (h) 2400 m2  (j) 2800 m2
      (k) 3000 m2 ',
     '(l) 3600 m2  (m) 4320 m2  (n) 4500 m2  (p) 4800 m2
      (q) 5400 m2 ',
     '(r) 6000 m2  (s) 7200 m2  (t) greater than 7200 m2
      ']).


xkb_question(area_sprinklered_Group_D,
     [' What is the building area ?',
      ' choose the closest number greater than the building
        area',
      '',
     '(a) 1600  m2  (b) 2000 m2  (c) 2400 m2  (d) 2500 m2
      (e) 3000 m2 ',
     '(f) 3200 m2   (g) 4000 m2  (h) 4800 m2  (j) 5600 m2
      (k) 6000 m2 ',
     '(l) 7200 m2   (m) 8640 m2  (n) 9000 m2  (p) 9600 m2
      (q) 10800 m2 ',
     '(r) 12000 m2  (s) 14400 m2  (t) greater than 14000 m2
        ']).


xkb_question(area_unsprinklered_Group_E,
     [' What is the building area ?',
      ' choose the closest number greater than the building
        area',
      '',
     '(a) 600  m2  (b) 750 m2   (c) 800 m2  (d) 900 m2  (e)
```

```
        1000 m2 ',
    '(f) 1200 m2  (g) 1250 m2  (h) 1500 m2 (j) greater than
        1500 m2 ']).


xkb_question(area_sprinklered_Group_E,
      [' What is the building area ?',
       ' choose the closest number greater than the building
         area',
       '',
    '(a) 1200  m2  (b) 1500 m2  (c) 1600 m2  (d) 1800 m2
     (e) 2000 m2 ',
    '(f) 2400 m2   (g) 2500 m2  (h) 3000 m2  (j) 3125 m2
     (k) 3600 m2 ',
    '(l) 3750 m2   (m) 4500 m2  (n) 4688 m2  (p) 4800 m2
     (q) 5000 m2 ',
    '(r) 5625 m2   (s) 6000 m2  (t) 6250 m2  (u) 7200 m2
     (v) 7500 m2',
    '(w) greater than 7500 m2 ']).


xkb_question(area_unsprinklered_Group_F_Div1,
      [' What is the building area ?',
       ' choose the closest number greater than the building
         area',
       '',
    '(a) 400  m2  (b) 500 m2   (c) 600 m2  (d) 750 m2  (e)
        800 m2 ',
    '(f) 900  m2  (g) 1000 m2  (h) 1200 m2 (j) 1500 m2 (k)
        1800 m2',
    '(l) 2400 m2  (m) 3000 m2  (n) 3600 m2 (p) greater than
        3600 m2']).


xkb_question(area_sprinklered_Group_F_Div1,
      [' What is the building area ?',
       ' choose the closest number greater than the building
         area',
       '',
    '(a) 800  m2  (b) 1000 m2  (c) 1200 m2  (d) 1500 m2
     (e) 1600 m2 ',
    '(f) 1800 m2  (g) 1875 m2  (h) 2000 m2  (j) 2250 m2
     (k) 2400 m2 ',
    '(l) 2500 m2  (m) 3000 m2  (n) 3600 m2  (p) 3750 m2
     (q) 4500 m2 ',
    '(r) 4800 m2  (s) 6000 m2  (t) 7200 m2  (u) 7500 m2
     (v) 9000 m2 ',
    '(w) greater than 9000 m2 ']).


xkb_question(area_unsprinklered_Group_F_Div2A,
```

```
      [' What is the building area ?',
       ' choose the closest number greater than the building
         area',
       '',
      '(a) 600  m2  (b) 750 m2   (c) 900 m2  (d) 1000 m2 (e)
           1250 m2 ',
      '(f) 1500 m2  (g) 1600 m2  (h) 2000 m2 (j) 2400 m2 (k)
           3000 m2',
      '(l) 3200 m2  (m) 3750 m2  (n) 4000 m2 (p) 4500 m2 (q)
           4800 m2',
      '(r) 5625 m2  (s) 6000 m2  (t) 6750 m2 (u) 7500 m2 (v)
           9000 m2',
      '(w) 11250 m2  (x) 13500 m2  (y) greater than 13500
            m2']).


xkb_question(area_unsprinklered_Group_F_Div2B,
      [' What is the building area ?',
       ' choose the closest number greater than the building
         area',
       '',
      '(a) 800 m2   (b) 1000 m2  (c) 1070 m2  (d) 1200 m2
       (e) 1340 m2 ',
      '(f) 1500 m2  (g) 1600 m2  (h) 1800 m2  (j) 1875 m2
       (k) 2000 m2 ',
      '(l) 2250 m2  (m) 2500 m2  (n) 2700 m2  (p) 2812 m2
       (q) 3000 m2 ',
      '(r) 3375 m2  (s) 3750 m2  (t) 4500 m2  (u) greater
           than 4500 m2']).

xkb_question(area_sprinklered_Group_F_Div2A,
      [' What is the building area ?',
       ' choose the closest number greater than the building
         area',
       '',
      '(a) 1200 m2  (b) 1500 m2  (c) 1800 m2 (d) 3000 m2 (e)
           3200 m2 ',
      '(f) 3750 m2  (g) 4000 m2  (h) 4500 m2 (j) 4800 m2 (k)
           6000 m2',
      '(l) 6400 m2  (m) 7500 m2  (n) 8000 m2 (p) 9000 m2 (q)
           9600 m2',
      '(r) 11250 m2 (s) 12000 m2 (t) 13500 m2 (u) 15000 m2
       (v) 18000 m2',
      '(w) 22500 m2  (x) 27000 m2  (y) greater than 27000
           m2']).


xkb_question(area_sprinklered_Group_F_Div2B,
      [' What is the building area ?',
       ' choose the closest number greater than the building
         area',
       '',
```

```
      '(a) 1600 m2  (b) 2000 m2  (c) 2140 m2  (d) 2400 m2
       (e) 2680 m2 ',
      '(f) 3000 m2  (g) 3200 m2  (h) 3600 m2  (j) 3750 m2
       (k) 4000 m2 ',
      '(l) 4500 m2  (m) 5000 m2  (n) 5400 m2  (p) 5624 m2
       (q) 6000 m2 ',
      '(r) 6750 m2  (s) 7500 m2  (t) 9000 m2  (u) greater
           than 9000 m2']).


xkb_question(area_unsprinklered_Group_F_Div3,
      [' What is the building area ?',
       ' choose the closest number greater than the building
         area',
       '',
      '(a) 800  m2  (b) 1000 m2  (c) 1200 m2 (d) 1500 m2 (e)
           1600 m2 ',
      '(f) 1800 m2  (g) 2000 m2  (h) 2400 m2 (j) 2880 m2 (k)
           3000 m2',
      '(l) 3600 m2  (m) 4320 m2  (n) 4500 m2 (p) 4800 m2 (q)
           5400 m2',
      '(r) 5600 m2  (s) 6000 m2  (t) 7000 m2 (u) 7200 m2 (v)
           8400 m2',
      '(w) 9000 m2  (x) 10800 m2  (y) greater than 10800
            m2'],.


xkb_question(area_sprinklered_Group_F_Div3,
      [' What is the building area ?',
       ' choose the closest number greater than the building
         area',
       '',
      '(a) 1600 m2  (b) 2000 m2  (c) 2400 m2  (d) 3000 m2
       (e) 3200 m2 ',
      '(f) 3600 m2  (g) 4000 m2  (h) 4800 m2  (j) 5760 m2
       (k) 6000 m2 ',
      '(l) 7200 m2  (m) 8640 m2  (n) 9000 m2  (p) 9600 m2
       (q) 10800 m2 ',
      '(r) 11200 m2  (s) 12000 m2  (t) 14000 m2  (u) 14400 m2
       (v) 16800 m2',
      '(w) 18000 m2  (x) 21600 m2  (y) greater than 21600
           m2']).


xkb_question(help,
 [' Select the term you wish to know the meaning',
  '',
  ' (a) Building     (b) Building area            (c)
        Building face ',
  ' (d) Combustible  (e) Combustible construction  (f)
         Exposing building face',
  ' (g) Fire compartment  (h) Fire damper  (i) Fire load',
```

```
    ' (j) Fire-protection rating (k) Fire-resistance (l)
        Fire-resistance rating',
    ' (m) Fire-retardant treated wood  (n) Fire separation
      (o) Fire stop',
    ' (p) Firewall              (q) Grade          (r) Heavy
        timber construction',
    ' (s) Limiting distance     (t) Mezzanine      (u)
        Noncombustible',
    ' (v) Sprinklered           (w) Storage garage (x) Storey
    '      ]).

xkb_question(licensed_beverage_establishment,
    [' Is the building used as a licensed beverage
       establishment ?']).

xkb_question(restaurant,
    [' Is the building used as a restaurant ?']).

xkb_question(school_or_college,
   [' Is the building used as a school or college ?']).

xkb_question(occupant_load,
   [' What is the occupant load of the building ?',
    ' i.e,the number of persons for which the building is
      designed',
    ' choose the closest number greater than the occupant
      load',
    ' ',
    ' (a) 10    (b) 25    (c) 25 above or below the first
        storey',
    ' (d) 40    (e) 75 above or below the first storey',
    ' (f) 100   (g) 150   (h) 150 above or below the first
        storey',
    ' (j) 300   (k) 300 below the seating area',
    ' (l) 500   (m) greater than 500 ']).


end_of_file.
```

```
/* fire_for.pl
 * knowledge base for fire regulations used for
 * forward chaining mechanism
 */

/* FIRE REGULATIONS */

/* Rule 12 */

xkb_fireregulation(
       [' Rule_12',
        ' Building shall be of combustible or noncombustible
          construction',
        ' used either singly or in combination,and ',
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
              45 min,',
        ' (b) mezzanines shall have,if of combustible
              construction, a fire-resistance rating of not
              less than 45 min,',
        ' (c) roof assemblies shall have,if of combustible
              construction, a fire-resistance rating of not
              less than 45 min,and',
        ' (d) all loadbearing walls,columns and arches
              supporting an assembly required to have a
              fire-resistance rating shall have a
              fire-resistance rating of not less than 45 min
              or shall be of noncombustible construction.'])
                              :-

                    xkb_identify(' Group A Division 1 '),
                       parmset(number_storeys,c,[a]),
                       prop(auditorium_floor_height),
                       parm(occupant_load_floor,c,a).


/* Rule 13 */

xkb_fireregulation(
       [' Rule_13',
        ' Building shall be of heavy timber or
          noncombustible construction used either singly or
          in combination,and',
       '  (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
              45 min, and',
       '  (b) all loadbearing walls,columns shall have a
              fire-resistance rating not less than that
              required for the supported assembly. ']) :-

                    xkb_identify(' Group A Division 1 '),
                       parmset(number_storeys,c,[a]),
```

```
                        parm(occupant_load_floor,c,b),
                        parm(building_area,c,1).


/* Rule 14 */

xkb_fireregulation(
     [' Rule_14',
      ' Building shall be of noncombustible construction ',
      '  (a) floor assemblies shall be fire separations with
          a fire-resistance rating of not less than 2 h,
      '  (b) mezzanines and roof assemblies shall have a
          fire resistance rating of not less than 1 h, ',
      '  (c) all loadbearing walls,columns and arches shall
          have a fire resistance rating not less than
          that required for the supported assembly.']) :-

                  xkb_identify(' Group A Division 1 '),
                      \+  known(fireregulation,_).


/* Rule 15 */
 xkb_fireregulation(
     [' Rule_15',
      ' Building shall be of combustible or noncombustible',
      ' construction used either singly or in combination.
              ']) :-
                  xkb_identify(' Group A Division 2 '),
                  parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),
                parmset(area_unsprinklered_Group_A,c,[a]))
           |  (parm(number_streets_building_faces,c,b),
               parmset(area_unsprinklered_Group_A,c,[a,b]))
           |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_A,c,[a,b,c]))))
         | ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),
                parmset(area_sprinklered_Group_A,c,[a]))
           |  (parm(number_streets_building_faces,c,b),
               parmset(area_sprinklered_Group_A,c,[a,b]))
           |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_A,c,[a,b,c])))))).


/* Rule 16 */

 xkb_fireregulation(
     [' Rule_16',
      ' Building shall be of combustible or noncombustible',
```

```
      ' construction used singly or in combination. ']) :-
                  xkb_identify(' Group A Division 2 '),
                  parmset(number_storeys,c,[a,b]),
                 \+ prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),
                 parmset(area_unsprinklered_Group_A,c,[a]))
             |  (parm(number_streets_building_faces,c,b),
                 parmset(area_unsprinklered_Group_A,c,[a,b]))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_A,c,[a,b,c]))),
                 \+ known(fireregulation,_).


/* Rule 17 */
xkb_fireregulation(
    [' Rule_17',
    ' Building shall be of combustible or noncombustible ',
     ' construction used either singly or in combination,
       and ',
     ' (a) floor assemblies shall be fire separations and,if
           of combustible construction,shall have a fire
           resistance rating of not less than 45 min,',
     ' (b) mezzanines shall have,if of combustible
           construction, a fire-resistance rating of not
           less than 45 min,',
     ' (c) roof assemblies shall have,if of combustible
           construction, a fire-resistance rating of not
           less than 45 min,',
     ' (d) all loadbearing walls,columns and arches
           supporting an assembly required to have a
           fire-resistance rating shall have a fire-',
     '     resistance rating of not less than 45 min or
           shall be of noncombustible construction.']) :-
                    xkb_identify(' Group A Division 2 '),
                       (( parmset(number_storeys,c,[a]),
       ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_A,c,[b,c,d,e,f,g,h,j]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_A,c,[c,d,e,f,g,h,j,k,l]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_A,c,[d,e,f,g,h,j,k,l,m])))))
         |  ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_A,c,[b,c,d,e,f,g,h,j]))
            |  (parm(number_streets_building_faces,c,b),
```

```
parmset(area_sprinklered_Group_A,c,[c,d,e,f,g,h,j,k,l]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_A,c,[d,e,f,g,h,j,k,l,m])))))))

              |           (  parmset(number_storeys,c,[b]),
       ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_A,c,[b,c,d]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_A,c,[c,d,e]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_A,c,[d,e,f])))))
         | ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),
                parmset(area_sprinklered_Group_A,c,[b,c,d]))
              |  (parm(number_streets_building_faces,c,b),
                parmset(area_sprinklered_Group_A,c,[c,d,e]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_A,c,[d,e,f])))))))).


/* Rule 20 */
xkb_fireregulation(
       [' Rule_20',
        ' Building shall be of noncombustible
          construction, and',
        ' (a) floor assemblies shall be fire separations
            with a fire-resistance rating of not less than
            1 h,',
        ' (b) mezzanines shall have a fire-resistance rating
            of not less than 1 h, and',
        ' (c) roof assemblies shall have a fire-resistance
            rating of not less than 1 h, and',
        ' (d) all loadbearing walls,columns and arches shall
            have a fire-resistance rating not less than
            that required for the supported assembly.'])
                  :-
                xkb_identify(' Group A Division 2 '),
                parmset(number_storeys,c,[a,b,c,d,e]),
                 \+ known(fireregulation,_).


/* Rule 22 */
xkb_fireregulation(
       [' Rule_22',
        ' Building shall be of noncombustible
```

```
        construction,and',
      ' (a) floor assemblies shall be fire separations
          with a fire-resistance rating of not less than
          2 h,',
      ' (b) mezzanines shall have a fire-resistance rating
          of not less than 1 h, and',
      ' (c) roof assemblies shall have a fire-resistance
          rating of not less than 1 h, and',
      ' (d) all loadbearing walls,columns and arches shall
          have a fire-resistance rating not less than
          that required for the supported assembly.'])
                :-
              xkb_identify(' Group A Division 2 '),
               \+ known(fireregulation,_).


/* Rule 23 */
xkb_fireregulation(
      [' Rule_23',
       ' Building shall be of combustible or noncombustible
         construction used either singly or in
         combination.']) :-

                  xkb_identify(' Group A Division 3 '),
                parmset(number_storeys,c,[a]),
     ((prop(unsprinklered),
            ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_A,c,[a,b,c,d,e]))
          |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_A,c,[a,b,c,d,e,f,g]))
          |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_A,c,[a,b,c,d,e,f,g,h])))))
       | ((\+ prop(unsprinklered)),
            ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_A,c,[a,b,c,d,e]))
          |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_A,c,[a,b,c,d,e,f,g]))
          |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_A,c,[a,b,c,d,e,f,g,h])))))).



/* Rule 24 */
```

```
xkb_fireregulation(
        [' Rule_24',
         ' Building shall be of combustible or noncombustible
           construction used either singly or in combination,
           and',
         ' (a) mezzanines shall have,if of combustible
               construction, a fire-resistance rating of not
               less than 45 min,',
         ' (b) roof assemblies shall have,if of combustible
               construction, a fire-resistance rating of not
               less than 45 min,',
         ' (c) all loadbearing walls,columns and arches
               supporting an assembly required to have a
               fire-resistance rating shall have a fire-',
         '     resistance  rating of not less than 45 min or
               shall be of noncombustible construction.']) :-
                     xkb_identify(' Group A Division 3 '),
                     parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_A,c,[f,g,h,j,k,l,m]))
             |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_A,c,[h,j,k,l,m,n,p]))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_A,c,[j,k,l,m,n,p,q])))))
         | ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_A,c,[f,g,h,j,k,l,m]))
             |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_A,c,[h,j,k,l,m,n,p]))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_A,c,[j,k,l,m,n,p,q])))))).



/* Rule 27 */
xkb_fireregulation(
        [' Rule_27',
         ' Building shall be of noncombustible
           construction, and',
         ' (a) floor assemblies shall be fire separations
               with a fire-resistance rating of not less than
               1 h,',
         ' (b) mezzanines shall have a  fire-resistance
               rating of not less than 1 h,',
```

```
        ' (c) roof assemblies shall have a fire-resistance
              rating of not less than 45 min or be of heavy
              timber construction, and',
        ' (d) all loadbearing walls,columns and arches shall
              have a fire-resistance rating not less than
              that required for the supported assembly,
              except that arches are permitted to be of
              heavy timber construction.']) :-
                    xkb_identify(' Group A Division 3 '),
                   (( parmset(number_storeys,c,[a]),
     ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_A,c,[n,p,q,r]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_A,c,[q,r,s]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_A,c,[r,s,t])))))
          |  ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_A,c,[n,p,q,r]))
            |  (parm(number_streets_building_faces,c,b),
                parmset(area_sprinklered_Group_A,c,[q,r,s]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_A,c,[r,s,t])))))))

         |    (parmset(number_storeys,c,[b]),
     (( prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_A,c,[a,b,c,d,e,f,g,h,j,k,l]
                    ))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_A,c,[a,b,c,d,e,f,g,h,j,k,l,
m,n]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_A,c,[a,b,c,d,e,f,g,h,j,k,l,
m,n,p])))))
 |  ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_A,c,[a,b,c,d,e,f,g,h,j,k,l])))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_A,c,[a,b,c,d,e,f,g,h,j,k,l,m,
n]))
```

```
            |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_A,c,[a,b,c,d,e,f,g,h,j,k,l,m,
n,p])))))))).



/*  Rule 28 */
xkb_fireregulation(
       [' Rule_28',
        ' Building shall be of noncombustible construction,
          and',
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
              2 h,',
        ' (b) mezzanines shall have a fire-resistance rating
              of not less than 1 h, and',
        ' (c) roof assemblies shall have a fire-resistance
              rating of not less than 1 h, and',
        ' (d) all loadbearing walls,columns and arches shall
              have a fire-resistance rating not less than
              that required for the supported assembly.'])
                    :-
                  xkb_identify(' Group A Division 3 '),
                  \+ known(fireregulation,_).



/* Rule 29 */
xkb_fireregulation(
         [' Rule_29',
          ' Building shall be of noncombustible
            construction  except that
            (a) roof assemblies may be of heavy ',
          '     timber construction, and ',
          ' (b) Building may be of combustible construction
              provided ',
          ' (i) occupant load is less than 1500 persons,
              and',
          ' (ii) building has a limiting distance of at
              least 6 m.']) :-
                xkb_identify(' Group A Division 4 ').


/* Rule 30 */
xkb_fireregulation(
       [' Rule_30',
        ' Building shall be of noncombustible
          construction, and',
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
```

```
            2 h,',
        ' (b) mezzanines shall have a fire-resistance rating
            of not less than 1 h, and',
        ' (c) roof assemblies shall have a fire-resistance
            rating of not less than 1 h, and',
        ' (d) all loadbearing walls,columns and arches shall
            have a fire-resistance rating not less than
            that required for the supported assembly.'])
                    :-
                xkb_identify(' Group B Division 1 ').


/* Rule 31 */
 xkb_fireregulation(
     [' Rule_31',
      ' Building shall be of combustible or noncombustible',
      ' construction used either singly or in
        combination.']) :-

                xkb_identify(' Group B Division 2 '),
                parmset(number_storeys,c,[a]),
                (( prop(unsprinklered),

parmset(area_unsprinklered_Group_B,c,[a]))
                |  ((\+ prop(unsprinklered)),

parmset(area_sprinklered_Group_B,c,[a]))).



/* Rule 32 */
xkb_fireregulation(
         [' Rule_32',
          ' Building shall be of combustible or
            noncombustible construction used either singly
            or in combination, and',
          ' (a) floor assemblies shall be  fire separations
              with a fire-resistance rating of not less
              than 45 min,',
          ' (b) mezzanines shall have,if of combustible
              construction, a  fire-resistance rating of
              not less than 45 min,',
          ' (c) roof assemblies shall have,if of combustible
              construction, a fire-resistance rating of
              not less than 45 min, and',
          ' (d) all loadbearing walls, columns and arches
              shall have a fire-resistance rating not less
              than that required for the supported
               assembly.']) :-
                    xkb_identify(' Group B Division 2 '),
```

```
                        ((parmset(number_storeys,c,[a]),
                         ((prop(unsprinklered),

parmset(area_unsprinklered_Group_B,c,[b,c]))
                        | ((\+ prop(unsprinklered)),

parmset(area_sprinklered_Group_B,c,[b,c])))))
                        | (parmset(number_storeys,c,[b]),
                          ((prop(unsprinklered),

parmset(area_unsprinklered_Group_B,c,[b]))
                         | ((\+ prop(unsprinklered)),

parmset(area_sprinklered_Group_B,c,[b])))))).



/* Rule 33 */
xkb_fireregulation(
         [' Rule_33',
          ' Building shall be of noncombustible
            construction, and',
          ' (a) floor assemblies shall be  fire separations
              with a fire-resistance rating of not less
              than 1 h,',
          ' (b) mezzanines shall have a  fire-resistance
              rating of not less than 1 h,',
          ' (c) roof assemblies shall have a fire-resistance
              rating of not less than 1 h',
          ' (d) all loadbearing walls,columns and arches
              shall have a fire-resistance rating not less
              than that required for the supported
              assembly.']) :-
                    xkb_identify(' Group B Division 2 '),
                        \+ prop(unsprinklered),
                        ((parmset(number_storeys,c,[a]),

parmset(area_sprinklered_Group_B,c,[d,e,f]))
                        | (parmset(number_storeys,c,[b]),

parmset(area_sprinklered_Group_B,c,[c,d,e]))
                        | (parmset(number_storeys,c,[c]),

parmset(area_sprinklered_Group_B,c,[a,b,c,d]))).



/* Rule 34  */
```

```
xkb_fireregulation(
        [' Rule_34',
         ' Building shall be of noncombustible
           construction, and',
         ' (a) floor assemblies shall be  fire separations
             with a fire-resistance rating of not less
             than 2 h,',
         ' (b) mezzanines shall have a  fire-resistance
             rating of not less than 1 h,',
         ' (c) roof assemblies shall have a fire-resistance
             rating of not less than 1 h',
         ' (d) all loadbearing walls,columns and arches
             shall have a fire-resistance rating not less
             than that required for the supported
             assembly.']) :-

                  xkb_identify(' Group B Division 2 '),
                 \+ known(fireregulation,_).


/* Rule 35 */
xkb_fireregulation(
       [' Rule_35',
        ' Building shall be of combustible or noncombustible
          construction used either singly or in combination,
          and',
        ' (a) floor assemblies shall be  fire separations
            with a fire-resistance rating of not less than
            45 min,',
        ' (b) mezzanines shall have ,if of combustible
            construction, a fire-resistance rating of not
            less than 45 min, and',
        ' (c) all loadbearing walls,columns and arches shall
            have a fire-resistance rating not less than
            that required for the supported assembly.'])
                  :-
                   xkb_identify(' Group C '),
                  (( parmset(number_storeys,c,[a]),
     ((prop(unsprinklered),
             ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k]))
       |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l,
m]))
       |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l,
m,n,p,q])))))
       |  ((\+ prop(unsprinklered)),
```

```
            ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l,m]
))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l,m,
n,p,q])))))))

         |    (parmset(number_storeys,c,[b]),
     (( prop(unsprinklered),
            ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_C,c,[a,b,c,d]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_C,c,[a,b,c,d,e,f]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_C,c,[a,b,c,d,e,f,g,h])))))
 | ((\+ prop(unsprinklered)),
            ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_C,c,[a,b,c,d]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_C,c,[a,b,c,d,e,f]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_C,c,[a,b,c,d,e,f,g,h])))))))
         |    (parmset(number_storeys,c,[c]),
     (( prop(unsprinklered),
            ((parm(number_streets_building_faces,c,a),
              parmset(area_unsprinklered_Group_C,c,[a]))
            |  (parm(number_streets_building_faces,c,b),
              parmset(area_unsprinklered_Group_C,c,[a,b]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_C,c,[a,b,c,d])))))
 | ((\+ prop(unsprinklered)),
            ((parm(number_streets_building_faces,c,a),
              parmset(area_sprinklered_Group_C,c,[a]))
            |  (parm(number_streets_building_faces,c,b),
              parmset(area_sprinklered_Group_C,c,[a,b]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_C,c,[a,b,c,d])))))))).
```

```
/* Rule 38 */
xkb_fireregulation(
        [' Rule_38',
         ' Building shall be of combustible or noncombustible
           construction used eitheR singly or in combination,
           and',
         ' (a) floor assemblies shall be  fire separations
               with a fire-resistance rating of not less than
               1 h,',
         ' (b) mezzanines shall have ,if of combustible
               construction, a fire-resistance rating of not
               less than 1 h,',
         ' (c) roof assemblies shall have a fire-resistance
               rating of not less than 1 h, and',
         ' (d) all loadbearing walls,columns and arches shall
               have a fire-resistance rating not less than
               that required for the supported assembly.'])
                              :-
                    xkb_identify(' Group C '),
                   (( parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_C,c,[l,m,n]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_C,c,[n,p,q,r]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_C,c,[r,s])))))
          |  ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),
                parmset(area_sprinklered_Group_C,c,[l,m,n]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_C,c,[n,p,q,r]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_C,c,[r,s]))))))

         |    (parmset(number_storeys,c,[b]),
     (( prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_C,c,[e,f,g]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_C,c,[g,h,j]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_C,c,[j,k])))))
 |  ((\+ prop(unsprinklered)),
```

```
                ((parm(number_streets_building_faces,c,a),
                  parmset(area_sprinklered_Group_C,c,[e,f,g]))
              |  (parm(number_streets_building_faces,c,b),
                  parmset(area_sprinklered_Group_C,c,[g,h,j]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_C,c,[j,k])))))
          |    (parmset(number_storeys,c,[c]),
     (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),
                  parmset(area_unsprinklered_Group_C,c,[b,c]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_C,c,[c,d,e]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_C,c,[e,f,g])))))
 | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),
                  parmset(area_sprinklered_Group_C,c,[b,c]))
              |  (parm(number_streets_building_faces,c,b),
                  parmset(area_sprinklered_Group_C,c,[c,d,e]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_C,c,[e,f,g])))))))).



/* Rule 41 */
xkb_fireregulation(
       [' Rule_41',
        ' Building shall be of combustible or noncombustible
          construction used either singly or in combination,
          and',
        ' (a) floor assemblies shall be  fire separations
              with a fire-resistance rating of not less than
              1 h,',
        ' (b) mezzanines shall have ,if of combustible
              construction, a fire-resistance rating of not
              less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
              rating of not less than 1 h, and',
        ' (d) all loadbearing walls,columns and arches shall
              have a fire-resistance rating not less than
              that required for the supported assembly.'])
                    :-
                  xkb_identify(' Group C '),
                 \+ prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),
                 parmset(area_sprinklered_Group_C,c,[a]))
             |  (parm(number_streets_building_faces,c,b),
```

```
                    parmset(area_sprinklered_Group_C,c,[a,b]))
                |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_C,c,[a,b,c,d]))).


/* Rule 44 */
xkb_fireregulation(
       [' Rule_44',
        ' Building shall be of noncombustible
          construction, and',
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
              1 h,',
        ' (b) mezzanines shall have a  fire-resistance
              rating of not less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
              rating of not less than 1 h, and',
        ' (d) all loadbearing walls,columns and arches shall
              have a fire-resistance rating not less than
              that required for the supported assembly.'])
                    :-
                   xkb_identify(' Group C '),
                  (( parmset(number_storeys,c,[a]),
     ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_C,c,[p,q,r,s,t,u,v,w,x,y]))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_C,c,[s,t,u,v,w,x,y]))
            |   (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_C,c,[t,u,v,w,x,y])))))
          |   ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_C,c,[p,q,r,s,t,u,v,w,x,y]))
            |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_C,c,[s,t,u,v,w,x,y]))
            |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_C,c,[t,u,v,w,x,y])))))))

         |     (parmset(number_storeys,c,[b]),
     (( prop(unsprinklered),
         ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_C,c,[h,j,k,l,m,n,p,q,r,s,t,
u,v,w,x]))
```

```
        |   (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_C,c,[k,l,m,n,p,q,r,s,t,u,v,
w,y]))
         |   (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_C,c,[l,m,n,p,q,r,s,t,u,v,w,
x,y])))))
    | ((\+ prop(unsprinklered)),
        ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_C,c,[h,j,k,l,m,n,p,q,r,s,t,u,
v,w,x]))
             |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_C,c,[k,l,m,n,p,q,r,s,t,u,v,w,
x,y]))
             |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_C,c,[l,m,n,p,q,r,s.t,u,v,w,x,
y]))))))

         |     (parmset(number_storeys,c,[c]),
     (( prop(unsprinklered),
        ((parm(number_streets_building_faces,c,a), .

parmset(area_unsprinklered_Group_C,c,[d,e,f,g,h,j,k,l,m,n,p,
q,r,s,t,u]))
       |   (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_C,c,[f,g,h,j,k,l,m,n,p,q,r,
s,t,u,v,w]))
       |   (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_C,c,[h,j,k,l,m,n,p,q,r,s,t,
u,v,w,x])))))
 | ((\+ prop(unsprinklered)),
        ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_C,c,[d,e,f,g,h,j,k,l,m,n,p,q,
r,s,t,u]))
     |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_C,c,[f,g,h,j,k,l,m,n,p,q,r,s,
t,u,v,w]))
      |   (parm(number_streets_building_faces,c,c),
parmset(area_sprinklered_Group_C,c,[h,j,k,l,m,n,p,q,r,s,t,u,
v,w,x])))))))

|     (parmset(number_storeys,c,[d]),
     (( prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),
```

```
parmset(area_unsprinklered_Group_C,c,[h,j,k,l,m,n,p,q,r]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_C,c,[k,l,m,n,p,q,r,s,t]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_C,c,[l,m,n,p,q,r,s,t,u,v]))
))
 | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_C,c,[h,j,k,l,m,n,p,q,r]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_C,c,[k,l,m,n,p,q,r,s,t]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_C,c,[l,m,n,p,q,r,s,t,u,v])))))
))
          |     (parmset(number_storeys,c,[e]),
     (( prop(unsprinklered),
           ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l,
m,n]))
       |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l,
m,n,p,q,r]))
       |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l,
m,n,p,q,r,s])))))
 | ((\+ prop(unsprinklered)),
        ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l,m,
n]))
       |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l,m,
n,p,q,r]))
       |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l,m,
n,p,q,r,s])))))))

|     (parmset(number_storeys,c,[f]),
     (( prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),
```

```
parmset(area_unsprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l]
))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l,
m,n,p]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l,
m,n,p,q,r])))))
  |  ((\+ prop(unsprinklered)),
                 ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_C c,[a,b,c,d,e,f,g,h,j,k,l]))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l,m,
n,p]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_C,c,[a,b,c,d,e,f,g,h,j,k,l,m,
n,p,q,r])))))))).


   /* Rule 46 */

    xkb_fireregulation(
        [' Rule 46',
         ' Building shall be of noncombustible construction,
           and',
         ' (a) floor assemblies shall be fire separations',
         '     resistance rating of not less than 2 h,',
         ' (b) mezzanines shall have a  fire-resistance
               rating of not less than 1 h,',
         ' (c) roof assemblies shall have a fire-resistance
               rating of not less than 1 h, and',
         ' (d) all loadbearing walls,columns and arches shall
               have a fire-resistance rating not less than
               that required for the supported assembly.'])
                      :-
                    xkb_identify(' Group C '),
                    \+ known(fireregulation,_).



/* Rule 48 */

xkb_fireregulation(
       [' Rule_48',
        ' Building shall be of combustible or noncombustible
          construction used either singly or in combination,
          and',
```

```
        ' (a) floor assemblies shall be fire separations
            and, if of combustible construction, shall
            have a fire-resistance rating of not less than
            45 min, and',
        ' (b) all loadbearing walls,columns and arches
            supporting an assembly required to have a
            fire-resistance rating shall have a
            fire-resistance rating of not less than 45 min
            or shall be of noncombustible construction.'])
                    :-
                xkb_identify(' Group D '),
                  (( parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),
                parmset(area_unsprinklered_Group_D,c,[a,b]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_D,c,[a,b,c,d]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_D,c,[a,b,c,d,e])))))
         | ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),
                parmset(area_sprinklered_Group_D,c,[a,b]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_D,c,[a,b,c,d]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_D,c,[a,b,c,d,e])))))))

      |  ( parmset(number_storeys,c,[b]),
      ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),
                parmset(area_unsprinklered_Group_D,c,[a]))
            |  (parm(number_streets_building_faces,c,b),
                parmset(area_unsprinklered_Group_D,c,[a,b]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_D,c,[a,b,c])))))
         | ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),
                parmset(area_sprinklered_Group_D,c,[a]))
            |  (parm(number_streets_building_faces,c,b),
                parmset(area_sprinklered_Group_D,c,[a,b]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_D,c,[a,b,c])))))))).
```

```
/* Rule 49 */

xkb_fireregulation(
       [' Rule_49',
        ' Building shall be of combustible or noncombustible
          construction used either singly or in combination,
          and',
        ' (a) floor assemblies shall be  fire separations
              and, if of combustible construction, shall
              have a fire-resistance rating of not less than
              45 min,',
        ' (b) mezzanines shall have , if of combustible
              construction, a fire-resistance rating of not
              less than 45 min,',
        ' (c) roof assemblies shall have, if of combustible
              construction, a fire-resistance rating of not
              less than 45 min,',
        ' (d) all loadbearing walls,columns and arches
              supporting an assembly required to have a
              fire-resistance rating shall have a fire-',
        '     resistance rating of not less than 45 min or
              shall be of noncombustible construction.']) :-
                                  xkb_identify(' Group D '),
                                                        ( (
parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_D,c,[c,d,e,f,g,h,j,k,l,m,n,
p]))
             |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_D,c,[e,f,g,h,j,k,l,m,n,p,q,
r]))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_D,c,[f,g,h,j,k,l,m,n,p,q,r,
s])))))
          |  ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_D,c,[c,d,e,f,g,h,j,k,l,m,n,p]
))
             |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_D,c,[e,f,g,h,j,k,l,m,n,p,q,r]
))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_D,c,[f,g,h,j,k,l,m,n,p,q,r,s]
))))))
```

```
          |     (parmset(number_storeys,c,[b]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_D,c,[b,c,d,e,f,g,h]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_D,c,[c,d,e,f,g,h,j,k]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_D,c,[d,e,f,g,h,j,k,l])))))
  | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_D,c,[b,c,d,e,f,g,h]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_D,c,[c,d,e,f,g,h,j,k]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_D,c,[d,e,f,g,h,j,k,l])))))))
          |     (parmset(number_storeys,c,[c]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_D,c,[a,b,c,d,e,f]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_D,c,[a,b,c,d,e,f,g]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_D,c,[a,b,c,d,e,f,g,h])))))
  | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),
            parmset(area_sprinklered_Group_D,c,[a,b,c,d,e,f]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_D,c,[a,b,c,d,e,f,g]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_D,c,[a,b,c,d,e,f,g,h])))))))).



/* Rule 51 */

xkb_fireregulation(
       [' Rule_51',
        ' Building shall be of noncombustible construction,
          and',
        ' (a) floor assemblies shall be fire separations
```

```
                with a fire-resistance rating of not less than
                1 h,',
          ' (b) mezzanines shall have a fire-resistance rating
                of not less than 1 h,',
          ' (c) roof assemblies shall have a fire-resistance
                rating of not less than 1 h, except that in
                buildings of 1 storey in building height this
                requirement is waived, and',
          ' (d) all loadbearing walls,columns and arches shall
                have a fire resistance rating not less than
                that required for the supported assembly.'])
                        :-
                                xkb_identify(' Group D '),
                                                        ( (
parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_D,c,[q,r,s,t]))
             |  (parm(number_streets_building_faces,c,b),
                 parmset(area_unsprinklered_Group_D,c,[s,t]))
             |  (parm(number_streets_building_faces,c,c),
                 parmset(area_unsprinklered_Group_D,c,[t]))))
         |  ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_D,c,[q,r,s,t]))
             |  (parm(number_streets_building_faces,c,b),
                 parmset(area_sprinklered_Group_D,c,[s,t]))
             |  (parm(number_streets_building_faces,c,c),
                 parmset(area_sprinklered_Group_D,c,[t])))))))

        |    (parmset(number_storeys,c,[b]),
     (( prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_D,c,[j,k,l,m,n,p,q,r,s]))
             |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_D,c,[l,m,n,p,q,r,s,t]))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_D,c,[m,n,p,q,r,s,t])))))
 | ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_D,c,[j,k,l,m,n,p,q,r,s]))
             |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_D,c,[l,m,n,p,q,r,s,t]))
             |  (parm(number_streets_building_faces,c,c),
```

```
parmset(area_sprinklered_Group_D,c,[m,n,p,q,r,s,t])))))
          |     (parmset(number_storeys,c,[c]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_D,c,[g,h,j,k,l,m,n,p]))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_D,c,[h,j,k,l,m,n,p,q,r]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_D,c,[j,k,l,m,n,p,q,r,s])))))
  | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_D,c,[g,h,j,k,l,m,n,p]))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_D,c,[h,j,k,l,m,n,p,q,r]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_D,c,[j,k,l,m,n,p,q,r,s])))))))

|     (parmset(number_storeys,c,[d]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j,k,l]
))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j,k,l,
m,n]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j,k,l,
m,n,p,q])))))
  ! ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j,k,l]))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j,k,l,m,
n]))
              |   (parm(number_streets_building_faces,c,c),^N

parmset(area_sprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j,k,l,m,
n,p,q])))))))
          |     (parmset(number_storeys,c,[e]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),
```

```
parmset(area_unsprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j,k,l]
))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j,k,l,
m])))))
  |  ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j,k,l]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j,k,l,m]
))))))

|     (parmset(number_storeys,c,[f]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_D,c,[a,b,c,d,e,f,g,h]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j,k]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j,k,l]
))))
  |  ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_D,c,[a,b,c,d,e,f,g,h]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j,k]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_D,c,[a,b,c,d,e,f,g,h,j,k,l]))
)))))).



/* Rule 52 */
```

```
xkb_fireregulation(
       [' Rule_52',
        ' Building shall be of noncombustible construction,
          and',
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
              2 h,',
        ' (b) mezzanines shall have a fire-resistance rating
              of not less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
              rating of not less than 1 h, except that in
              buildings of 1 storey in building height this
              requirement is waived, and',
        ' (d) all loadbearing walls,columns and arches shall
              have a fire resistance rating not less than
              that required for the supported assembly.'])
                        :-

                              xkb_identify(' Group D '),
                               \+ known(fireregulation,_).


/* Rule 53 */

xkb_fireregulation(
       [' Rule_53,
        ' Building shall be of combustible or noncombustible
          construction used either singly or in combination,
          and',
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
              45 min, and',
        ' (b) all loadbearing walls,columns and arches shall
              have a fire-resistance rating not less than
              that required for the supported assembly.'])
                     :-
                  xkb_identify(' Group E '),
                    (( parmset(number_storeys,c,[a]),
     ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_E,c,[a,b,c,d,e]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_E,c,[a,b,c,d,e,f,g]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_E,c,[a,b,c,d,e,f,g,h])))))
        | ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),
```

```
parmset(area_sprinklered_Group_E,c,[a,b,c,d,e]))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_E,c,[a,b,c,d,e,f,g]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_E,c,[a,b,c,d,e,f,g,h])))))))

       |   ( parmset(number_storeys,c,[b]),
       ((prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),
                  parmset(area_unsprinklered_Group_E,c,[a]))
              |   (parm(number_streets_building_faces,c,b),
                  parmset(area_unsprinklered_Group_E,c,[a,b]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_E,c,[a,b,c,d])))))
           | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),
                  parmset(area_sprinklered_Group_E,c,[a]))
              |   (parm(number_streets_building_faces,c,b),
                  parmset(area_sprinklered_Group_E,c,[a,b]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_E,c,[a,b,c,d])))))))).
```

```
/* Rule 54 */
xkb_fireregulation(
        [' Rule_54',
         ' Building shall be of combustible or noncombustible
           construction used either singly or in combination,
           and',
         ' (a) floor assemblies shall be fire separations
               with a fire-resistance rating of not less than
               45 min,',
         ' (b) mezzanines shall have,if of combustible
               construction, a fire-resistance rating of not
               less than 45 min,',
         ' (c) roof assemblies shall have a fire-resistance
               rating of not less than 45 min, except that in
               buildings not more than 1 storey in building
               height, the fire-resistance rating is
               permitted to be waived provided the roof
               assembly is of noncombustible construction or
               is constructed as a fire-retardant treated',
         '     wood roof system,',
         ' (d) all loadbearing walls,columns and arches
               supporting an assembly required to have a
```

```
            fire-resistance rating shall have a fire-',
          '     resistance rating of not less than 45 min or
            shall be of noncombustible construction,
            except that such members and',
          '     assemblies supporting a fire separation shall
            have a fire-resistance rating not less than
            that required for the supported assembly.'])
                                :-
                                 xkb_identify(' Group E '),
                           (( parmset(number_storeys,c,[a]),

      ((prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_E,c,[f,g,h]))
             |  (parm(number_streets_building_faces,c,b),
                 parmset(area_unsprinklered_Group_E,c,[h]))
             |  (parm(number_streets_building_faces,c,c),
                 parmset(area_unsprinklered_Group_E,c,[h])))))
         |  ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_E,c,[f,g,h,j,k,l,m,n,p]))
             |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_E,c,[h,j,k,l,m,n,p,q,r,s]))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_E,c,[j,k,l,m,n,p,q,r,s,t,u]))
))))

         |     (parmset(number_storeys,c,[b]),
     (( prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_E,c,[b,c,d,e,f]))
             |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_E,c,[c,d,e,f,g,h]))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_E,c,[e,f,g,h])))))
 |  ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_E,c,[b,c,d,e,f]))
             |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_E,c,[c,d,e,f,g,h]))
             |  (parm(number_streets_building_faces,c,c),
                 parmset(area_sprinklered_Group_E,c,[j])))))))
         |     (parmset(number_storeys,c,[c]),
```

```
     (( prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_E,c,[a,b,c]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_E,c,[a,b,c,d,e]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_E,c,[a,b,c,d,e,f]))))
 | ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),
                parmset(area_sprinklered_Group_E,c,[a,b,c]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_E,c,[a,b,c,d,e]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_E,c,[a,b,c,d,e,f])))))).



/* Rule 56 */

xkb_fireregulation(
       [' Rule_56',
        ' Building shall be of noncombustible construction,
          and',
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
              2 h,',
        ' (b) mezzanines shall have a fire-resistance
              rating of not less than 1 h,' ,
        ' (c) roof assemblies shall have a fire-resistance
              rating of not less than 1 h, and',
        ' (d) all loadbearing walls,columns and arches shall
              have a fire resistance rating not less than
              that required for the supported assembly.'])
                    :-
                            xkb_identify(' Group E '),
                                                    ( (
parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),
                parmset(area_unsprinklered_Group_E,c,[h]))
            |  (parm(number_streets_building_faces,c,b),
                parmset(area_unsprinklered_Group_E,c,[h]))
            |  (parm(number_streets_building_faces,c,c),
                parmset(area_unsprinklered_Group_E,c,[h]))))
        |  ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),
```

```
parmset(area_sprinklered_Group_E,c,[q,r,s,t,u,v,w]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_E,c,[t,u,v,w]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_E,c,[v,w])))))))

          |     (parmset(number_storeys,c,[b]),
     (( prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),
                 parmset(area_unsprinklered_Group_E,c,[h]))
              |  (parm(number_streets_building_faces,c,b),
                parmset(area_unsprinklered_Group_E,c,[h]))
              |  (parm(number_streets_building_faces,c,c),
                 parmset(area_unsprinklered_Group_E,c,[h]))))
 | ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_E,c,[g,h,j,k,l,m,n,p,q,r,s,t,
u,v]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_E,c,[j,k,l,m,n,p,q,r,s,t,u,v,
w]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_E,c,[l,m,n,p,q,r,s,t,u,v,w]))
))))
          |     (parmset(number_storeys,c,[c]),
     (( prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_E,c,[d,e,f,g,h]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_E,c,[f,g,h]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_E,c,[g,h])))))
 | ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_E,c,[d,e,f,g,h,j,k,l,m,p,q]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_E,c,[f,g,h,j,k,l,m,n,p,q,r,s,
t]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_E,c,[g,h,j,k,l,m,n,p,q,r,s,t,
```

```
u,v])))))))

|     (parmset(number_storeys,c,[d]),
          (((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_E,c,[a,b,c,d,e,f,g,h,j,k,l]))
               |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_E,c,[a,b,c,d,e,f,g,h,j,k,l,m,
n]))
               |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_E,c,[a,b,c,d,e,f,g,h,j,k,l,m,
n,p,q,r])))))))
          |     (parmset(number_storeys,c,[e]),
          (((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_E,c,[a,b,c,d,e,f,g,h]))
               |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_E,c,[a,b,c,d,e,f,g,h,j,k,l]))
               |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_E,c,[a,b,c,d,e,f,g,h,j,k,l,m]
))))))

          |     (parmset(number_storeys,c,[f]),
               (((\+ prop(unsprinklered)),
                 ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_E,c,[a,b,c,d,e,f,g]))
               |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_E,c,[a,b,c,d,e,f,g,h,j]))
               |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_E,c,[a,b,c,d,e,f,g,h,j,k,l]))
)))))).


/* Rule 57   */
   xkb_fireregulation(
       [' Rule_57',
        ' Building shall be of noncombustible
          construction, and',
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
              3 h,',
        ' (b) mezzanines shall have a  fire-resistance
              rating of not less than 1.5 h,',
```

```
        ' (c) roof assemblies shall have a  fire-resistance
            rating of not less than 1.5 h',
        ' (d) all loadbearing walls,columns and arches shall
            have a fire resistance rating not less than
            that required for the supported assembly.'])
                        :-
                              xkb_identify(' Group E '),
                               \+ prop(unsprinklered),
                            \+ known(fireregulation,_).


/* Rule 58 */

xkb_fireregulation(
       [' Rule_58',
        ' Building shall be of combustible or noncombustible
          construction used either singly or in combination,
          and',
        ' (a) floor assemblies shall be  fire separations
            and if of combustible construction shall have
            a  fire-resistance rating of not less than 45
            min, and',
        ' (b) all loadbearing walls,columns and arches
            supporting an assembly required to have a
            fire-resistance rating shall have a fire-',
        '     resistance rating of not less than 45 min or
            shall be of noncombustible construction.']) :-
                  xkb_identify(' Group F Division 1 '),
                                                    ( (
parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div1,c,[a,b,c,d,e]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div1,c,[a,b,c,d,e,f,g]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div1,c,[a,b,c,d,e,f,g,h])
)))
         | ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div1,c,[a,b,c,d,e]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div1,c,[a,b,c,d,e,f,g,h]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div1,c,[a,b,c,d,e,f,g,h,j,k
```

```
]))))))

      |   ( parmset(number_storeys,c,[b]),
      ((prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div1,c,[a]))
             |   (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div1,c,[a,b]))
             |   (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div1,c,[a,b,c])))))
         | ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),
                 parmset(area_sprinklered_Group_F_Div1,c,[a]))
             |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div1,c,[a,b]))
             |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div1,c,[a,b,c])))))))).



/* Rule 59 */

xkb_fireregulation(
       [' Rule_59',
        ' Building shall be of heavy timber or
          noncombustible construction used either singly or
          in combination, and',
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
              45 min, and',
        ' (b) all loadbearing walls,columns and arches shall
              have a fire-resistance rating not less than
              that required for the supported assembly.'])
                        :-
                   xkb_identify(' Group F Division 1 '),
                                \+ prop(unsprinklered),
                                                    ( (
parmset(number_storeys,c,[a]),
             ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div1,c,[f,g,h,j,k]))
             |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div1,c,[j,k,l,m]))
             |   (parm(number_streets_building_faces,c,c),
```

```
parmset(area_sprinklered_Group_F_Div1,c,[l,m,n])))))

          |     (parmset(number_storeys,c,[b]),
             ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div1,c,[b,c]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div1,c,[c,d]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div1,c,[d,e,f])))))
          |     (parmset(number_storeys,c,[c]),
             ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),
                  parmset(area_sprinklered_Group_F_Div1,c,[a]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div1,c,[a,b]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div1,c,[a,b,c])))))).



/* Rule 60 */

xkb_fireregulation(
       [' Rule_60',
        ' Building shall be of noncombustible
          construction, and',
        ' (a) floor assemblies shall be  fire separations
              with a fire-resistance rating of not less than
              2 h,',
        ' (b) mezzanines shall have a fire-resistance
              rating of not less than 1 h,',
        ' (c) roof assemblies shall have a  fire-resistance
              rating of not less than 1 h, and',
        ' (d) all loadbearing walls,columns and arches shall
              have a fire resistance rating not less than
              that required for the supported assembly.'])
                          :-
                    xkb_identify(' Group F Division 1 '),
                                                    ( (
parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div1,c,[f,g,h,j,k,l]))
```

```
                | (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div1,c,[h,j,k,l,m]))
                | (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div1,c,[j,k,l,m,n])))))
            | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div1,c,[f,g,h,j,k,l,m,n,p,q
,r]))
                | (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div1,c,[j,k,l,m,n,p,q,r,s])
)
                | (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div1,c,[l,m,n,p,q,r,s,t])))
)))

          |     (parmset(number_storeys,c,[b]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div1,c,[b,c,d,e,f,g,h]))
                | (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div1,c,[c,d,e,f,g,h,j]))
                | (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div1,c,[d,e,f,g,h,j,k])))
)
  | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div1,c,[b,c,d,e,f,g,h,j,k])
)
                | (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div1,c,[c,d,e,f,g,h,j,k,l,m
]))
                | (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div1,c,[d,e,f,g,h,j,k,l,m,n
])))))))
          |     (parmset(number_storeys,c,[c]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div1,c,[a,b,c,d,e]))
                | (parm(number_streets_building_faces,c,b),
```

```
parmset(area_unsprinklered_Group_F_Div1,c,[a,b,c,d,e,f,g]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div1,c,[a,b,c,d,e,f,g,h])
)))
  | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div1,c,[b,c,d,e]))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div1,c,[c,d,e,f,g,h]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div1,c,[d,e,f,g,h,j,k])))))
)
          |     (parmset(number_storeys,c,[d]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div1,c,[a,b,c]))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div1,c,[a,b,c,d]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div1,c,[a,b,c,d,e,f])))))
  | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div1,c,[a,b,c]))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div1,c,[a,b,c,d]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div1,c,[a,b,c,d,e,f])))))))
.



/* Rule 61  */

xkb_fireregulation(
        [' Rule_61',
         'Building shall be of noncombustible construction,
          and',
         ' (a) floor assemblies shall be  fire separations
               with a fire-resistance rating of not less than
               3 h,',
         ' (b) mezzanines shall have a fire-resistance
               rating of not less than 1.5 h,',
```

```
        ' (c) roof assemblies shall have a  fire-resistance
            rating of not less than 1.5 h, and',
        ' (d) all loadbearing walls,columns and arches shall
            have a fire-resistance rating not less than
            that required for the supported assembly.'])
                    :-
                xkb_identify(' Group F Division 1 '),
                        \+ prop(unsprinklered),
                                                ( (
parmset(number_storeys,c,[a]),
            ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),
              parmset(area_sprinklered_Group_F_Div1,c,[s]))
          |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div1,c,[t,u]))
          |   (parm(number_streets_building_faces,c),

parmset(area_sprinklered_Group_F_Div1,c,[u,v])))))

        |     (parmset(number_storeys,c,[b]),
          ((\+ prop(unsprinklered)),
            ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div1,c,[l,m]))
          |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div1,c,[n,p]))
          |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div1,c,[p,q])))))
        |     (parmset(number_storeys,c,[c]),
          ((\+ prop(unsprinklered)),
            ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div1,c,[f,g,h]))
          |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div1,c,[j,k,l]))
          |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div1,c,[l,m])))))

 |     (parmset(number_storeys,c,[d]),
            ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),
              parmset(area_sprinklered_Group_F_Div1,c,[d]))
          |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div1,c,[e,f,g]))
          |   (parm(number_streets_building_faces,c,c),
```

```
parmset(area_sprinklered_Group_F_Div1,c,[g,h,j])))))).



/* Rule 62 */

xkb_fireregulation(
        [' Rule_62',
         ' Building shall be of combustible or noncombustible
           construction used either singly or in combination,
           and',
         ' (a) floor assemblies shall be fire separations
               and if of combustible construction, shall have
               a  fire resistance rating of not less than 45
               min, and',
         ' (b) all loadbearing walls,columns and arches
               supporting an assembly required to have a
               fire-resistance rating shall have a
               fire-resistance rating of not less than 45 min
               or shall be of noncombustible construction.'])
                   :-
                  xkb_identify(' Group F Division 2 '),
                    (( parmset(number_storeys,c,[a]),.
      ((prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2A,c,[a,b,c,d]))
             |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2A,c,[a,b,c,d,e]))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2A,c,[a,b,c,d,e,f])))))
         |  ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2A,c,[a,b,c,d]))
             !  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2A,c,[a,b,c,d,e,f]))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2A,c,[a,b,c,d,e,f,g,h]))
))))

      |  ( parmset(number_storeys,c,[b]),
      ((prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2A,c,[a]))
             |  (parm(number_streets_building_faces,c,b),
```

```
parmset(area_unsprinklered_Group_F_Div2A,c,[a,b]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2A,c,[a,b,c])))))
         |  ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2A,c,[a]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2A,c,[a,b]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2A,c,[a,b,c])))))))).



/* Rule 63 */

xkb_fireregulation(
       [' Rule_63',
        ' Building shall be of combustible or noncombustible
          construction used either singly or in combination,
          and',
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
              45 min,',
        ' (b) mezzanines shall have ,if of combustible
              construction, a fire-resistance rating of not
              less than 45 min,',
        ' (c) roof assemblies shall have ,if of combustible
              construction, a fire-resistance rating of not
              less than 45 min, except that in',
        '     buildings not more than 1 storey in building
              height, the fire-resistance rating is
              permitted to be waived provided that the',
        '     roof assembly is constructed as a
              fire-retardant treated wood, and',
        ' (d) all loadbearing walls,columns and arches
              supporting an assembly required to have a
              fire-resistance rating shall have a fire-',
        '     resistance rating of not less than 45 min or
              shall be of noncombustible construction,
              except that such members and assemblies
              supporting a fire separation shall have a
              fire-resistance rating not less than that
              required for the supported assembly.']) :-
                  xkb_identify(' Group F Division 2 '),
                                                        ( (
parmset(number_storeys,c,[a]),
     ((prop(unsprinklered),
```

```
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2A,c,[e,f,g,h,j,k,l]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2A,c,[f,g,h,j,k,l,m,n]
))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2A,c,[g,h,j,k,l,m,n,p,
q])))))
           |  ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2A,c,[e,f,g,h,j,k,l]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2A,c,[g,h,j,k,l,m,n]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2A,c,[j,k,l,m,n,p,q])))))
))

          |     (parmset(number_storeys,c,[b]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2A,c,[b,c,d,e,f,g]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2A,c,[c,d,e,f,g,h]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2A,c,[d,e,f,g,h,j])))))
  | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2A,c,[b,c,d,e]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2A,c,[c,d,e,f,g]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2A,c,[d,e,f,g,h,j])))))))
          |     (parmset(number_storeys,c,[c]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2B,c,[a,b,c]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2B,c,[a,b,c,d,e]))
```

```
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2B,c,[a,b,c,d,e,f,g]))
))
 | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2B,c,[a,b,c]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2B,c,[a,b,c,d,e]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2B,c,[a,b,c,d,e,f,g])))))
))
          |    (parmset(number_storeys,c,[d]),
     (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2B,c,[a]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2B,c,[a,b]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2B,c,[a,b,c,d])))))
 | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2B,c,[a]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2B,c,[a,b]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2B,c,[a,b,c,d])))))))).



/* Rule 65 */

xkb_fireregulation(
       [' Rule_65',
        ' Building shall be of noncombustible
          construction, and',
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
              1 h,',
        ' (b) mezzanines shall have a fire-resistance
              rating of not less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
              rating of not less than 1 h, and',
```

```
        ' (d) all loadbearing walls,columns and arches shall
              have a fire-resistance rating not less than
              that required for the supported assembly.'])
                        :-
                    xkb_identify(' Group F Division 2 '),
                                                        ( (
parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2A,c,[m,n,p,q,r,s]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2A,c,[p,q,r,s,t,u]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2A,c,[r,s,t,u,v])))))
          |  ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2A,c,[m,n,p,q,r,s]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2A,c,[p,q,r,s,t,u]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2A,c,[r,s,t,u,v])))))))

         |     (parmset(number_storeys,c,[b]),
     (( prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2A,c,[h,j,k]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2A,c,[j,k,l,m]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2A,c,[k,l,m,n,p])))))
 |  ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2A,c,[h,j,k]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2A,c,[j,k,l,m]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2A,c,[k,l,m,n,p])))))))
         |     (parmset(number_storeys,c,[c]),
     (( prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),
```

```
parmset(area_unsprinklered_Group_F_Div2B,c,[d,e,f,g,h,j,k]))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2B,c,[f,g,h,j,k,l,m]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2B,c,[h,j,k,l,m,n,p,q]
))))
  | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2B,c,[d,e,f,g,h,j,k]))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2B,c,[f,g,h,j,k,l,m]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2B,c,[h,j,k,l,m,n,p,q]))
))))
          |     (parmset(number_storeys,c,[d]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2B,c,[b,c,d,e,f]))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2B,c,[c,d,e,f,g,h,j]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2B,c,[e,f,g,h,j,k,l]))
))
  | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2B,c,[b,c,d,e,f]))
              |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2B,c,[c,d,e,f,g,h,j]))
              |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2B,c,[e,f,g,h,j,k,l])))))
))).



/* Rule 66 */

xkb_fireregulation(
        [' Rule_66',
         ' Building shall be of noncombustible construction,
           and',
```

```
        ' (a) floor assemblies shall be  fire separations
            with a fire-resistance rating of not less than
            2 h,',
        ' (b) mezzanines shall have a  fire-resistance
            rating of not less than 1 h,',
        ' (c) roof assemblies shall have a  fire-resistance
            rating of not less than 1 h, and',
        ' (d) all loadbearing walls,columns and arches shall
            have a fire-resistance rating not less than
            that required for the supported assembly.'])
                    :-
                xkb_identify(' Group F Division 2 '),
                                                    ( (
parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2A,c,[t,u,v]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2A,c,[v,w]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2A,c,[w,x])))))
          |  ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2A,c,[t,u,v]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2A,c,[v,w]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2A,c,[w,x])))))))

         |    (parmset(number_storeys,c,[b]),
     (( prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2A,c,[l,m,n,p]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2A,c,[n,p,q,r]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2A,c,[q,r,s,t])))))
 |  ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2A,c,[l,m,n,p]))
            |  (parm(number_streets_building_faces,c,b),
```

```
parmset(area_sprinklered_Group_F_Div2A,c,[n,p,q,r]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2A,c,[q,r,s,t])))))
          |     (parmset(number_storeys,c,[c]),
     (( prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2B,c,[l,m,n,p,q]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2B,c,[n,p,q,r,s]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2B,c,[r,s,t])))))
 | ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2B,c,[l,m,n,p,q]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2B,c,[n,p,q,r]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2B,c,[r,s,t])))))
          |     (parmset(number_storeys,c,[d]),
     (( prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2B,c,[g,h,j,k,l]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2B,c,[k,l,m,n,p]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2B,c,[m,n,p,q,r])))))
 | ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2B,c,[g,h,j,k,l]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2B,c,[k,l,m,n,p]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2B,c,[m,n,p,q,r])))))))

 |     (parmset(number_storeys,c,[e]),
     (( prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2B,c,[a,b,c,d,e,f,g,h]
```

```
))
                |   (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2B,c,[a,b,c,d,e,f,g,h,
j,k,l]))
                |   (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2B,c,[a,b,c,d,e,f,g,h,
j,k,l,m,n])))))
  | ((\+ prop(unsprinklered)),
                  ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2B,c,[a,b,c,d,e,f,g,h]))
                |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2B,c,[a,b,c,d,e,f,g,h,j,
k,l]))
                |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2B,c,[a,b,c,d,e,f,g,h,j,
k,l,m,n])))))))
            |     (parmset(number_storeys,c,[f]),
      (( prop(unsprinklered),
                  ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div2B,c,[a,b,c,d,e,f]))
                |   (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div2B,c,[a,b,c,d,e,f,g,h,
j]))
                |   (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div2B,c,[a,b,c,d,e,f,g,h,
j,k,l])))))
  | ((\+ prop(unsprinklered)),
                  ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div2B,c,[a,b,c,d,e,f]))
                |   (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div2B,c,[a,b,c,d,e,f,g,h,j]
))
                |   (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div2B,c,[a,b,c,d,e,f,g,h,j,
k,l])))))))).
```

```
/* Rule 67  */
```

```
xkb_fireregulation(
       [' Rule_67',
        'Building shall be of noncombustible construction,
         and',
        ' (a) floor assemblies shall be  fire separations
              with a fire-resistance rating of not less than
              3 h,',
        ' (b) mezzanines shall have a  fire-resistance
              rating of not less than 1.5 h,',
        ' (c) roof assemblies shall have a  fire-resistance
              rating of not less than 1.5 h, and',
        ' (d) all loadbearing walls,columns and arches shall
              have a fire-resistance rating not less than
              that required for the supported assembly.'])
                        :-
                  xkb_identify(' Group F Division 2 '),
                       \+  known(fireregulation,_).



/* Rule 68 */

xkb_fireregulation(
       [' Rule_68',
        ' Building shall be of combustible or noncombustible
          construction used either singly or in combination,
          and',
        ' (a) floor assemblies shall be fire separations
              and, if of combustible construction, shall
              have a  fire-resistance rating of not less
              than 45 min, and',
        ' (b) all loadbearing walls,columns and arches
              supporting an assembly required to have a
              fire-resistance rating shall have a  fire-',
        '     resistance rating of not less than 45 min or
              shall be of noncombustible construction.']) :-
                    xkb_identify(' Group F Division 3 '),
                                                        ( (
parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c,d,e]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h])
)))
```

```
          | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h,j,k
]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h,j,k
,l])))))

       |  ( parmset(number_storeys,c,[b]),
       ((prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div3,c,[a]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c])))))
          | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),
                  parmset(area_sprinklered_Group_F_Div3,c,[a]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div3,c,[a,b]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div3,c,[a,b,c])))))))).



/* Rule 69 */

xkb_fireregulation(
       [' Rule_69',
        ' Building shall be of combustible or noncombustible
          construction used either singly or in combination,
          and',
        ' (a)  floor assemblies shall be fire separations
               and, if of combustible construction, shall
               have a  fire-resistance rating of not less
               than 45 min,',
        ' (b)  mezzanines shall have, if of combustible
               construction, a  fire-resistance rating of
               not less than 45 min,',
        ' (c)  roof assemblies shall have, if combustible
```

```
                construction, a  fire-resistance rating of
                not less than 45 min, except that in
                buildings not more than 1 storey in building
                height, the fire-resistance rating is
                permitted to be waived provided that the roof
                assembly is constructed as a fire-retardant',
          '     treated wood roof system,',
          ' (d) all loadbearing walls,columns and arches
                supporting an assembly required to have a
                fire-resistance rating shall have a fire-',
          '     resistance rating of not less than 45 min or
                shall be of noncombustible construction.'])
                          :-
                     xkb_identify(' Group F Division 3 '),
                                                        ( (
parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div3,c,[f,g,h,j,k,l,m,n,p
]))
             |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div3,c,[h,j,k,l,m,n,p,q,r
,s]))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div3,c,[j,k,l,m,n,p,q,r,s
,t,u])))))
          |  ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div3,c,[j,k,l,m,n,p]))
             |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div3,c,[l,m,n,p,q,r,s]))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div3,c,[m,n,p,q,r,s,t,u])))
)))

         |    (parmset(number_storeys,c,[b]),
     (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div3,c,[b,c,d,e,f,g,h]))
             |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div3,c,[c,d,e,f,g,h,j,k])
)
             |  (parm(number_streets_building_faces,c,c),
```

```
parmset(area_unsprinklered_Group_F_Div3,c,[d,e,f,g,h,j,k,l])
)))
 | ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div3,c,[b,c,d,e,f,g,h]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div3,c,[c,d,e,f,g,h,j,k]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div3,c,[d,e,f,g,h,j,k,l])))
)))
         |    (parmset(number_storeys,c,[c]),
     (( prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c,d,e]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h])
)))
 | ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div3,c,[a,b,c,d,e]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h])))
)))
         |    (parmset(number_storeys,c,[d]),
     (( prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c,d]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c,d,e,f])))))
 | ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div3,c,[a,b,c]))
            |  (parm(number_streets_building_faces,c,b),
```

```
parmset(area_sprinklered_Group_F_Div3,c,[a,b,c,d]))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div3,c,[a,b,c,d,e,f])))))))
.



 /* Rule 71 */

xkb_fireregulation(
      [' Rule_71',
        ' Building shall be of heavy timber or
        noncombustible construction used either singly or
        in combination.']) :-

                        xkb_identify(' Group F Division 3 '),
                                                          ( (
parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div3,c,[q,r]))
             |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div3,c,[t]))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div3,c,[v])))))
          |  ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div3,c,[q,r]))
             |  (parm(number_streets_building_faces,c,b),
                parmset(area_sprinklered_Group_F_Div3,c,[t]))
             |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div3,c,[v])))))))).



/*  Rule 72 */

xkb_fireregulation(
       [' Rule_72',
        ' Building shall be of noncombustible
        construction.']) :-
                 xkb_identify(' Group F Division 3 '),
                              parmset(number_storeys,c,[a]),
                               prop(low_fire_load).
```

```
/* Rule 73 */
xkb_fireregulation(
       [' Rule_73',
        ' Building used as a storage garage and having no
          other occupancy above it may have its floor, wall,
          ceiling and roof assemblies constructed without a
          fire-resistance rating, and of noncombustible
          construction.']) :-
                    xkb_identify(' Group F Division 3 '),
                          prop(storage_garage),
                                                    \ +
parmset(area_unsprinklered_Group_F_Div3,c,[y]),
                              \+ parmset(number_storeys,c,[g]).


/* Rule 74 */

xkb_fireregulation(
       [' Building shall be of noncombustible construction,
          and',
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance r  .ng of not less than
              1 h,',
        ' (b) mezzanines shall have a fire-resistance
              rating of not less than 1 h,',
        ' (c) roof assemblies shall have a  fire-resistance
              rating of not less than 1 h, and',
        ' (d) all loadbearing walls,columns and arches shall
              have a fire-resistance rating not less than
              that required for the supported assembly.'])
                        :-
                    xkb_identify(' Group F Division 3 '),
                                                    ( (
parmset(number_storeys,c,[a]),
      ((prop(unsprinklered),
              ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div3,c,[s,t,u,v,w,x,y]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div3,c,[u,v,w,x,y]))
            |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div3,c,[w,x,y])))))
          |  ((\+ prop(unsprinklered)),
              ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div3,c,[s,t,u,v,w,x,y]))
            |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div3,c,[u,v,w,x,y]))
```

```
                | (parm(number_streets_building_faces,c,c),
parmset(area_sprinklered_Group_F_Div3,c,[w,x,y])))))))

          |     (parmset(number_storeys,c,[b]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),
parmset(area_unsprinklered_Group_F_Div3,c,[j,k,l,m,n,p,q,r,s
,t,u]))
                | (parm(number_streets_building_faces,c,b),
parmset(area_unsprinklered_Group_F_Div3,c,[l,m,n,p,q,r,s,t,u
,v,w]))
                | (parm(number_streets_building_faces,c,c),
parmset(area_unsprinklered_Group_F_Div3,c,[m,n,p,q,r,s,t,u,v
,w,x])))))
  | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),
          parmset(area_sprinklered_Group_F_Div3,c,[s,t,u]))
                | (parm(number_streets_building_faces,c,b),
parmset(area_sprinklered_Group_F_Div3,c,[l,m,n,p,q,r,s,t,u,v
,w]))
                | (parm(number_streets_building_faces,c,c),
parmset(area_sprinklered_Group_F_Div3,c,[m,n,p,q,r,s,t,u,v,w
,x])))))))
          |     (parmset(number_storeys,c,[c]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),
parmset(area_unsprinklered_Group_F_Div3,c,[f,g,h,j,k,l,m,n,p
]))
                | (parm(number_streets_building_faces,c,b),
parmset(area_unsprinklered_Group_F_Div3,c,[h,j,k,l,m,n,p,q,r
,s]))
                | (parm(number_streets_building_faces,c,c),
parmset(area_unsprinklered_Group_F_Div3,c,[j,k,l,m,n,p,q,r,s
,t,u])))))
  | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),
parmset(area_sprinklered_Group_F_Div3,c,[f,g,h,j,k,l,m,n,p])
)
                | (parm(number_streets_building_faces,c,b),
parmset(area_sprinklered_Group_F_Div3,c,[h,j,k,l,m,n,p,q,r,s
]))
```

```
                | (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div3,c,[j,k,l,m,n,p,q,r,s,t
,u])))))))
          |    (parmset(number_storeys,c,[d]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div3,c,[d,e,f,g,h,j,k,l])
)
                | (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div3,c,[e,f,g,h,j,k,l,m,n
]))
                | (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div3,c,[g,h,j,k,l,m,n,p,q
])))))
  | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div3,c,[d,e,f,g,h,j,k,l]))
                | (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div3,c,[e,f,g,h,j,k,l,m,n])
)
                | (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div3,c,[g,h,j,k,l,m,n,p,q])
)))))

  |    (parmset(number_storeys,c,[e]),
      (( prop(unsprinklered),
                ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h,j
]))
                | (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h,j
,k,l]))
                | (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h,j
,k,l,m])))))
  | ((\+ prop(unsprinklered)),
                ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h,j])
)
                | (parm(number_streets_building_faces,c,b),
```

```
parmset(area_sprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h,j,k
,l]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h,j,k
,l,m])))))))
          |     (parmset(number_storeys,c,[f]),
     (( prop(unsprinklered),
               ((parm(number_streets_building_faces,c,a),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h])
)
              |  (parm(number_streets_building_faces,c,b),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h,j
,k]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_unsprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h,j
,k,l])))))
 | ((\+ prop(unsprinklered)),
               ((parm(number_streets_building_faces,c,a),

parmset(area_sprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h]))
              |  (parm(number_streets_building_faces,c,b),

parmset(area_sprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h,j,k
]))
              |  (parm(number_streets_building_faces,c,c),

parmset(area_sprinklered_Group_F_Div3,c,[a,b,c,d,e,f,g,h,j,k
,l])))))))).



/* Rule 75 */

xkb_fireregulation(
       [' Building shall be of noncombustible construction,
          and',
        ' (a) floor assemblies shall be  fire separations
              with a fire-resistance rating of not less than
              2 h, except that such floor assemblies are
              permitted to be reduced to fire separations',
        '     with a fire-resistance rating of not less than
              1 h in a storage garage with all storeys
              constructed as open-air storeys,',
        ' (b) mezzanines shall have a  fire-resistance
              rating of not less than 1 h,',
        ' (c) roof assemblies shall have a  fire-resistance
              rating of not less than 1 h, and',
```

```
        ' (d) all loadbearing walls,columns and arches shall
              have a fire-resistance rating not less than
              that required for the supported assembly.'])
                        :-
                   xkb_identify(' Group F Division 3 '),
                              \+ known(fireregulation,_).


end_of_file.
```

```
/* show.pl */
 *
 * Displays fire regulations
 */

     fireclause(12,
        ([' Building shall be of combustible or
            noncombustible construction used either singly
            or in combination,and ',
         ' (a) floor assemblies shall be fire separations
         '       with a  fire-resistance rating of not less
                 than 45 min,',
         '  (b)  mezzanines shall have,if of combustible
                 construction, a fire-resistance rating of not
                 less than 45 min,',
         ' (c) roof assemblies shall have,if of combustible
                 construction,a fire-resistance rating of not
                 less than 45 min,and',
         '  (d)  all loadbearing walls,columns and arches
                 supporting an assembly required to have a
                 fire-resistance rating shall have a
                 fire-resistance rating of not less than 45
                 min or shall be of noncombustible
                 construction.']).

fireclause(13,
        ([' Building shall be of heavy timber or
            noncombustible construction used either singly
            or in combination,and
         '  (a) floor assemblies shall be fire separations
                 with a fire resistance rating of not less
                 than 45 min,and',
         '    (b) all loadbearing walls,columns shall have a
                 fire-resistance rating not less than that
                 required for the supported assembly. ']).

fireclause(14,
         ([' Building shall be of noncombustible
             construction ',
       '  (a) floor assemblies shall be fire separations with
             a fire-resistance rating of not less than 2 h,
       '  (b) mezzanines and roof assemblies shall have a
             fire resistance rating of not less than 1 h, ',
       '  (c) all loadbearing walls,columns and arches shall
             have a fire resistance rating not less than
             that required for the supported assembly.']).

fireclause(15,
      ([' Building shall be of combustible or
         noncombustible construction used either singly or
         in combination.']).
```

```
fireclause(16,
       (['Building shall be of combustible or
          noncombustible construction used singly or in
          combination.']).

fireclause(17,
     (['Building shall be of combustible or noncombustible
        construction used either singly or in combination,
        and ',
     ' (a) floor assemblies shall be fire separations and,if
          of combustible construction,shall have a fire
          resistance rating of not less than 45 min,',
     ' (b) mezzanines shall have,if of combustible
          construction,a fire-resistance rating of not less
          than 45 min,',
     ' (c) roof assemblies shall have,if of combustible
          construction,a fire-resistance rating of not less
          than 45 min,',
     ' (d) all loadbearing walls,columns and arches
          supporting an assembly required to have a
          fire-resistance rating shall have a fire-',
     '       resistance rating of not less than 45 min or
          shall be of noncombustible construction.']).

fireclause(18,
       (['Building shall be of noncombustible
          construction,and',
        ' (a) floor assemblies shall be fire separations
             with a fire-resistance rating of not less than
              1 h,',
        ' (b) mezzanines shall have a fire-resistance rating
             of not less than 1 h, and',
        ' (c) roof assemblies shall have a fire-resistance
             rating of not less than 1 h, and',
        ' (d) all loadbearing walls,columns and arches shall
             have a fire-resistance rating not less than
             that required for the supported assembly.']).


fireclause(19,
       ([' Building shall be of noncombustible
          construction,and',
        ' (a) floor assemblies shall be fire separations
             with a fire-resistance rating of not less than
              2 h,',
        ' (b) mezzanines shall have a fire-resistance rating
             of not less than 1 h, and',
        ' (c) roof assemblies shall have a fire-resistance
             rating of not less than 1 h, and',
        ' (d) all loadbearing walls,columns and arches shall
             have a fire-resistance rating not less than
      that required for the supported assembly.']).
```

```
fireclause(20,
      ([' Building shall be of combustible or noncombustible
          construction used either singly or in
          combination.']).

fireclause(21,
      ([' Building shall be of combustible or noncombustible
          construction used either singly or in combination,
          and',
        ' (a) mezzanines shall have,if of combustible
            construction, a fire-resistance rating of not
            less than 45 min,',
        ' (b) roof assemblies shall have,if of combustible
            construction,a fire-resistance rating of not
            less than 45 min,',
        ' (c) all loadbearing walls,columns and arches
            supporting an assembly required to have a
            fire-resistance rating shall have a fire-',
        '     resistance  rating of not less than 45 min or
            shall be of noncombustible construction.']).

fireclause(22,
      ([' Building shall be of noncombustible
          construction,and',
        ' (a) floor assemblies shall be fire separations
            with a fire-resistance rating of not less than
            1 h,',
        ' (b) mezzanines shall have a  fire-resistance
            rating of not less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
            rating of not less than 45 min or be of heavy
            timber construction,and',
        ' (d) all loadbearing walls,columns and arches shall
            have a fire-resistance rating not less than
            that required for the supported assembly,
            except that arches are permitted to be of
            heavy timber construction.']).

  fireclause(23,
     ([' Building shall be of noncombustible
          construction,and',
        ' (a) floor assemblies shall be fire separations
            with a fire-resistance rating of not less than
             2 h,',
        ' (b) mezzanines shall have a fire-resistance rating
            of not less than 1 h, and',
        ' (c) roof assemblies shall have a fire-resistance
            rating of not less than 1 h, and',
        ' (d) all loadbearing walls,columns and arches shall
            have a fire-resistance rating not less than
            that required for the supported assembly.']).
```

```
fireclause(24,
        (['  Building shall be of noncombustible
             construction  except that (a) roof assemblies
             may be of heavy timber construction, and ',
           ' (b) Building may be of combustible construction
             provided ',
           ' (i) occupant load is less than 1500 persons,
             and',
           ' (ii) building has a limiting distance of at
                least 6m.']).

fireclause(25,
     ([    '  Building  shall  be  of  noncombustible
          construction,and',
         ' (a) floor assemblies shall be fire separations
             with a fire-resistance rating of not less than
             2 h,',
         ' (b) mezzanines shall have a fire-resistance rating
             of not less than 1 h, and',
         ' (c) roof assemblies shall have a fire-resistance
             rating of not less than 1 h, and',
         ' (d) all loadbearing walls,columns and arches shall
             have a fire-resistance rating not less than
             that required for the supported assembly.']).

fireclause(26,
    (['  Building shall be of combustible or noncombustible',
       '   construction  used  either  singly  or  in
          combination.']).


fireclause(27,
      ([   ' Building shall be of combustible or
             noncombustible construction used either singly
             or in combination, and',
           ' (a) floor assemblies shall be  fire separations
             with a fire-resistance rating of not less
             than 45 min,',
           ' (b) mezzanines shall have,if of combustible
              construction,',
           '    a  fire-resistance rating of not less than 45
             min,',
           ' (c) roof assemblies shall have,if of combustible
             construction, a fire-resistance rating of not
             less than 45 min,and',
           '  (d)  all  loadbearing  walls,columns  and  arches
             shall have a fire-resistance rating not less
             than  that  required  for  the  supported
             assembly.']).

fireclause(28,
      ([  ' Building shall be of noncombustible
```

```
            construction,and',
          ' (a) floor assemblies shall be  fire separations
              with a fire-resistance rating of not less
              than 1 h,',
          ' (b) mezzanines shall have a  fire-resistance
              rating of not less than 1 h,',
          ' (c) roof assemblies shall have a fire-resistance
              rating of not less than 1 h',
          ' (d) all loadbearing walls,columns and arches
              shall have a fire-resistance rating not less
              than that required for the supported
              assembly.']).

fireclause(29,
      ([ ' Building shall be of noncombustible
            construction,and',
          ' (a) floor assemblies shall be  fire separations
              with a fire-resistance rating of not less
              than 2 h,',
          '  (b)  mezzanines shall have a  fire-resistance
              rating of not less than 1 h,',
          ' (c) roof assemblies shall have a fire-resistance
              rating of not less than 1 h',
          ' (d) all loadbearing walls,columns and arches
              shall have a fire-resistance rating not less
              than  that  required  for  the  supported
                  assembly.']).

fireclause(30,
    ([   ' Building shall be of combustible or noncombustible
           construction used either singly or in
           combination,and',
         ' (a) floor assemblies shall be  fire separations
             with a fire-resistance rating of not less than
             45 min,',
         ' (b) mezzanines shall have ,if of combustible
             construction,',
         '     a fire-resistance rating of not less than 45
              min,and',
         ' (c) all loadbearing walls,columns and arches shall
             have a fire-resistance rating not less than
             that required for the supported assembly.']).

fireclause(31,
    ([   ' Building shall be of combustible or noncombustible
           construction  used  either  singly  or  in
           combination,and',
         ' (a) floor assemblies shall be  fire separations
             with a fire-resistance rating of not less than
             1 h,',
         '  (b)  mezzanines shall have ,if of combustible
             construction,a fire-resistance rating of not
```

```
            less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
            rating of not less than 1 h,and',
        ' (d) all loadbearing walls,columns and arches shall
            have a fire-resistance rating not less than
            that required for the supported assembly.']).

fireclause(32,
     ([ ' Building shall be of combustible or noncombustible
          construction used either singly or in
          combination,and',
        ' (a) floor assemblies shall be fire separations
            with a fire-resistance rating of not less than
            1 h,',
        '  (b) mezzanines shall have ,if of combustible
            construction,a fire-resistance rating of not
            less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
            rating of not less than 1 h,and',
        ' (d) all loadbearing walls,columns and arches shall
            have a fire-resistance rating not less than
            that required for the supported assembly.']).

fireclause(33,
     ([    '   Building shall be of noncombustible
          construction,and',
        ' (a) floor assemblies shall be fire separations
            with a fire-resistance rating of not less than
             1 h,',
        '  (b) mezzanines shall have a fire-resistance
            rating of not less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
             rating of not less than 1 h,and',
        ' (d) all loadbearing walls,columns and arches shall
            have a fire-resistance rating not less than
            that required for the supported assembly.']).

fireclause(34,
    ([   ' Building shall be of noncombustible
          construction,and',
        ' (a) floor assemblies shall be fire separations
            with a fire-resistance rating of not less than
            2 h,',
        ' (b) mezzanines shall have a fire-resistance
            rating of not less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
            rating of not less than 1 h,and',
        ' (d) all loadbearing walls,columns and arches shall
            have a fire-resistance rating not less than
            that required for the supported assembly.']).

fireclause(35,
```

```
    ([  ' Building shall be of combustible or noncombustible
          construction used either singly or in
          combination,and',
        ' (a) floor assemblies shall be fire separations
              and,if of combustible construction, shall have
              a fire-resistance rating of not less than 45
              min, and',
        ' (b) all loadbearing walls,columns and arches
              supporting an assembly required to have a
              fire-resistance rating shall have a
              fire-resistance rating of not less than 45 min
                or   shall   be   of   noncombustible
               construction.']).

fireclause(36,
     ([ ' Building shall be of combustible or noncombustible
          construction used either singly or in combination,
          and',
        ' (a) floor assemblies shall be fire separations
              and,if of combustible construction, shall have
              a fire-resistance rating of not less than 45
              min,',
        ' (b) mezzanines shall have , if of combustible
      construction,a fire-resistance rating of not
        less than 45 min,',
        ' (c) roof assemblies shall have, if of combustible
              construction, a fire-resistance rating of not
              less than 45 min,',
        ' (d) all loadbearing walls,columns and arches
              supporting an assembly required to have a
              fire-resistance rating shall have a fire-',
        '     resistance rating of not less than 45 min or
              shall be of noncombustible construction.']).

fireclause(37,
     ([ ' Building shall be of noncombustible
          construction,and',
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
              1 h,',
        ' (b) mezzanines shall have a fire-resistance rating
              of not less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
              rating of not less than 1 h,except that in
              buildings of 1 storey in building height this
              requirement is waived,and',
        ' (d) all loadbearing walls,columns and arches shall
              have a fire resistance rating not less than
              that required for the supported assembly.']).

fireclause(38,
     ([   '   Building   shall   be   of   noncombustible
```

```
        construction,and',
        ' (a) floor assemblies shall be fire separations
            with a fire-resistance rating of not less than
            2 h,',
        ' (b) mezzanines shall have a fire-resistance rating
            of not less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
            rating of not less than 1 h,except that in
            buildings of 1 storey in building height this
            requirement is waived,and',
        ' (d) all loadbearing walls,columns and arches shall
            have a fire- resistance rating not less than
            that required for the supported assembly.']).

fireclause(39,
    ([ ' Building shall be of combustible or noncombustible
        construction used either singly or in combination,
        and',
        ' (a) floor assemblies shall be fire separations
            with a fire-resistance rating of not less than
            45 min,and',
        ' (b) all loadbearing walls,columns and arches shall
            have a fire-resistance rating not less than
            that required for the supported assembly.']).

fireclause(40,
    ([  ' Building shall be of combustible or noncombustible
        construction used either singly or in
        combination,and',
        ' (a) floor assemblies shall be fire separations
            with a fire-resistance rating of not less than
            45 min,',
        ' (b) mezzanines shall have,if of combustible
            construction,a fire-resistance rating of not
            less than 45 min,',
        ' (c) roof assemblies shall have a fire-resistance
            rating of not less than 45 min,except that
      in buildings not more than 1 storey in
  building height,the fire-resistance rating          is
permitted to be waived provided the roof
assembly is of noncombustible construction
or is constructed as a fire-retardant
treated wood roof system,',
        ' (d) all loadbearing walls,columns and arches
            supporting an assembly required to have a
            fire-resistance rating shall have a fire-',
        '     resistance rating of not less than 45 min or
            shall be of noncombustible construction,except
            that such members and assemblies supporting a
            fire separation shall have a fire-resistance
            rating not less than that required for the',
        '     supported assembly.']).
```

```
fireclause(41,
        ' Building shall be of noncombustible
          construction,and',
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
               2 h,',
        ' (b) mezzanines shall have a fire-resistance
               rating of not less than 1 h,' ,
        ' (c) roof assemblies shall have a fire-resistance
              rating of not less than 1 h,and',
        ' (d) all loadbearing walls,columns and arches shall
              have a fire resistance rating not less than
              that required for the supported assembly.']).

fireclause(42,
        '    Building    shall    be    of    noncombustible
          construction,and',
        '  (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
               3 h,',
        '  (b) mezzanines shall have a   fire-resistance
              rating of not less than 1.5 h,',
        ' (c) roof assemblies shall have a  fire-resistance
              rating of not less than 1.5 h',
        ' (d) all loadbearing walls,columns and arches shall
              have a fire resistance rating not less than
              that required for the supported assembly.']).

fireclause(43,
     ([ ' Building shall be of combustible or noncombustible
          construction    used    either    singly    or    in
          combination,and',
        '  (a) floor assemblies shall be  fire separations
              and if of combustible construction shall have
              a  fire-resistance rating of not less than 45
              min,and',
        ' (b) all loadbearing walls,columns and arches
              supporting an assembly required to have a
              fire-resistance rating shall have a fire-',
        '     resistance rating of not less than 45 min or
              shall be of noncombustible construction.']).

fireclause(44,
    ([      ' Building shall be of heavy timber or
          noncombustible construction used either singly or
          in combination,and',
        '  (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less than
              45 min,and',
        ' (b) all loadbearing walls,columns and arches shall
              have a fire-resistance rating not less than
              that required for the supported assembly.']).
```

```
fireclause(45,
     ([    ' Building shall be of noncombustible
           construction,and',
        ' (a) floor assemblies shall be fire separations
           with a fire-resistance rating of not less than
           2 h,',
        ' (b) mezzanines shall have a fire-resistance
           rating of not less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
           rating of not less than 1 h,and',
        ' (d) all loadbearing walls,columns and arches shall
           have a fire- resistance rating not less than
           that required for the supported assembly.']).

fireclause(46,
     ([       'Building shall be of noncombustible
           construction,and',
        ' (a) floor assemblies shall be fire separations
           with a fire-resistance rating of not less than
           3 h,',
        '  (b) mezzanines shall have a fire-resistance
           rating of not less than 1.5 h,',
        ' (c) roof assemblies shall have a fire-resistance
           rating of not less than 1.5 h, and',
        ' (d) all loadbearing walls,columns and arches shall
           have a fire-resistance rating not less than
           that required for the supported assembly.']).

fireclause(47,
     ([ ' Building shall be of combustible or noncombustible
           construction used either singly or in
           combination,and',
        ' (a) floor assemblies shall be fire separations
           and, if of combustible construction, shall
           have a fire- resistance rating of not less
           than 45 min,and',
        ' (b) all loadbearing walls,columns and arches
           supporting an assembly required to have a
           fire-resistance rating shall have a ',
        '     fire-resistance rating of not less than 45 min
           or shall be of noncombustible
           construction.']).

fireclause(48,
     ([ ' Building shall be of combustible or noncombustible
           construction used either singly or in
           combination,and',
        ' (a) floor assemblies shall be fire separations
           with a fire-resistance rating of not less than
           45 min,',
        ' (b) mezzanines shall have ,if of combustible
```

```
            construction, a fire-resistance rating of not
            less than 45 min,',
        ' (c) roof assemblies shall have ,if of combustible
            construction, a fire-resistance rating of not
            less than 45 min,except that in buildings not
            more than 1 storey in building height,the
            fire-resistance rating is permitted to be
            waived provided that the roof assembly is
            constructed as a fire-retardant treated      '
            wood,and',
        ' (d) all loadbearing walls,columns and arches
            supporting an assembly required to have a
            fire-resistance rating shall have a fire-',
        '     resistance rating of not less than 45 min or
            shall be of noncombustible construction,except
            that such members and assemblies supporting a
            fire separation shall have a fire-resistance
            rating not less than that required for the ',
        '     supported assembly.']).


fireclause(49,
     ([ ' Building shall be of noncombustible
          construction,and',
        ' (a) floor assemblies shall be fire separations
            with a fire-resistance rating of not less than
             1 h,',
        ' (b) mezzanines shall have a fire-resistance
             rating of not less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
            rating of not less than 1 h,and',
        ' (d) all loadbearing walls,columns and arches shall
            have a fire-resistance rating not less than
            that required for the supported assembly.']).

fireclause(50,
     ([    '   Building   shall   be   of   noncombustible
          construction,and',
        ' (a) floor assemblies shall be  fire separations
            with a fire-resistance rating of not less than
            2 h,',
        ' (b) mezzanines shall have a  fire-resistance
            rating of not less than 1 h,',
        ' (c) roof assemblies shall have a  fire-resistance
            rating of not less than 1 h,and',
        ' (d) all loadbearing walls,columns and arches shall
            have a fire-resistance rating not less than
            that required for the supported assembly.']).

fireclause(51,
      ([    'Building   shall   be   of   noncombustible
          construction,and',
```

```
        ' (a) floor assemblies shall be fire separations
              with a fire-resistance rating of not less
              than 3 h,',
        ' (b) mezzanines shall have a fire-resistance
              rating of not less than 1.5 h,',
        ' (c) roof assemblies shall have a fire-resistance
              rating of not less than 1.5 h,and',
        ' (d) all loadbearing walls,columns and arches shall
              have a fire-resistance rating not less than
              that required for the supported assembly.']).

fireclause(52,
    ([ ' Building shall be of combustible or noncombustible
         construction used either singly or in
         combination,and',
       ' (a) floor assemblies shall be fire separations
             and, if of combustible construction, shall
             have a fire-resistance rating of not less
             than 45 min,and',
       ' (b) all loadbearing walls,columns and arches
             supporting an assembly required to have a
             fire-resistance rating shall have a fire-',
       '     resistance rating of not less than 45 min or
             shall be of noncombustible construction.']).

fireclause(53,
    ([ ' Building shall be of combustible or noncombustible
         construction used either singly or in
         combination,and',
       ' (a) floor assemblies shall be fire separations
             and, if of combustible construction, shall
             have a fire-resistance rating of not less
             than 45 min,',
       ' (b) mezzanines shall have, if of combustible
             construction,a fire-resistance rating of not
             less than 45 min,',
       ' (c) roof assemblies shall have, if combustible
             construction, a fire-resistance rating of
             not less than 45 min,except that in buildings
             not more than 1 storey in building height,',
       '     the fire-resistance rating is permitted to be
             waived provided that the roof assembly is
             constructed as a fire-retardant treated wood
             roof system,',
       ' (d) all loadbearing walls,columns and arches
             supporting an assembly required to have a
             fire-resistance rating shall have a fire-',
       '     resistance rating of not less than 45 min or
             shall be of noncombustible construction.']).

fireclause(54,
    ([ ' Building shall be of heavy timber or
```

```
        noncombustible construction used either singly or
        in combination.']).

fireclause(55,
     ([    '   Building   shall   be   of   noncombustible
        construction.']).

fireclause(56,
     ([ ' Building used as a storage garage and having no
        other occupancy above it may have its floor,wall,
        ceiling and roof assemblies constructed without a
        fire-resistance rating,and of noncombustible
        construction.']).


fireclause(57,
      (['    Building   shall   be   of   noncombustible
        construction,and',
        ' (a) floor assemblies shall be fire separations
             with a fire-resistance rating of not less
             than 1 h,',
        '  (b) mezzanines shall have a fire-resistance
             rating of not less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
             rating of not less than 1 h,and',
        ' (d) all loadbearing walls,columns and arches shall
             have a fire-resistance rating not less than
             that required for the supported assembly.']).


fireclause(58,
      (['    Building   shall   be   of   noncombustible
        construction,and',
        ' (a) floor assemblies shall be  fire separations
             with a fire-resistance rating of not less
             than 2 h,except that such floor assemblies
             are permitted to be reduced to fire
             separations with a fire-resistance rating of
             not less than 1 h in a storage garage with
             all storeys constructed as open-air
             storeys,',
        '  (b) mezzanines shall have a  fire-resistance
             rating of not less than 1 h,',
        ' (c) roof assemblies shall have a fire-resistance
             rating of not less than 1 h,and',
        ' (d) all loadbearing walls,columns and arches shall
             have a fire-resistance rating not less than
             that required for the supported assembly.']).


r(A) :- fireclause(A,FIREID1),
        writeln(FIREID1).
```

```
end_of_file.
```

```
/* Firealarm.pl  */

/* Fire Alarm and Detection Systems */

:- retractall(xkb_firealarm(_)).


/* Rule 56 */

xkb_firealarm(
    [' Fire Alarm is not required ']) :-
                              xkb_identify(' Group C '),
                            (\+ (prop(hotel_or_motel)
                                | prop(school_or_college))),
                              ( prop(exit)
                              | ((parm(number_storeys,c,a)
                               | parm(number_storeys,c,b)
                                                              |
parm(number_storeys,c,c)),prop(access))).


/* Rule 57 */

xkb_firealarm(
    [' Fire Alarm is not required ']) :-
                              xkb_identify(' Group C '),
                              prop(hotel_or_motel),
                             ( parm(number_storeys,c,a)
                             | parm(number_storeys,c,b)
                             | parm(number_storeys,c,c)),
                               prop(suite_access).

/* Rule 58 */

xkb_firealarm(
    [' Fire Alarm System is required. ',
     ' Heat detectors shall be installed in every room.',
     '  Smoke  detectors  shall  be  installed  in  every
        corridor.'])                                  :-
                    xkb_identify(' Group A Division 1 '),
                  (\+ (prop(licensed_beverage_establishment)
                       | prop(restaurant)
                       | prop(school_or_college))),
                  parmset(occupant_load,c,[j,l,m]).

 /* Rule 59 */

xkb_firealarm(
    [' Fire Alarm System is required ']) :-
                     ( xkb_identify(' Group A Division 2 ')
               | xkb_identify(' Group A Division 3 ')),
               ( \+ (prop(licensed_beverage_establishment)
```

```
                    | prop(restaurant)
                    | prop(school_or_college))),
                   parmset(occupant_load,c,[j,l,m]).


/* Rule 60 */

xkb_firealarm(
    [' Fire Alarm System is required ']) :-
                     xkb_identify(' Group A Division 2 '),
                    ( prop(licensed_beverage_establishment)
                    | prop(restaurant)),
                      parmset(occupant_load,c,[g,j,l,m]).


/* Rule 61 */

xkb_firealarm(
    [' Fire Alarm System is required ']) :-
                     xkb_identify(' Group A Division 2 '),
                     prop(school_or_college),
                    parmset(occupant_load,c,[e,f,g,h,j,l,m]).


/* Rule 62 */

xkb_firealarm(
    [' Fire Alarm System is required ']) :-
                     xkb_identify(' Group A Division 4 '),
                     parmset(occupant_load,c,[l,m]).


/* Rule 63 */

xkb_firealarm(
    [' Fire Alarm System is required. ',
     ' Heat detectors shall be installed in every room.',
     ' Smoke detectors shall be installed in every room',
     ' and corridor.']) :-
                    ( xkb_identify(' Group B Division 1 ')
                    | xkb_identify(' Group B Division 2 ')),

parmset(occupant_load,c,[b,c,d,e,f,g,h,j,l,m]).


/* Rule 64 */

xkb_firealarm(
    [' Fire Alarm System is required. ',
     ' Smoke detectors shall be installed in every public
        corridor.']) :-
                     xkb_identify(' Group C '),
```

```
                    (parm(number_storeys,c,a)
                     | parm(number_storeys,c,b)
                     | parm(number_storeys,c,c)),

parmset(occupant_load,c,[b,c,d,e,f,g,h,j,l,m]).


/* Rule 65 */

xkb_firealarm(
    [' Fire Alarm System is required. ',
     ' Heat detectors shall be installed in every suite
       which is not a dwelling unit.',
     ' Smoke detectors shall be installed in every public
       corridor.']) :-
                     xkb_identify(' Group C '),
                   (\+ (parm(number_storeys,c,a)
                     | parm(number_storeys,c,b)
                     | parm(number_storeys,c,c))),

parmset(occupant_load,c,[b,c,d,e,f,g,h,j,l,m]).


/* Rule 66 */

xkb_firealarm(
    [' Fire Alarm System is required ']) :-
                     xkb_identify(' Group D '),
                     parmset(occupant_load,c,[l,m]).


/* Rule 67 */

xkb_firealarm(
    [' Fire Alarm System is required ']) :-
                     xkb_identify(' Group E '),
                    ( parmset(occupant_load,c,[l,m])
                  | (\+ (parm(number_storeys,c,a)
                      | parm(number_storeys,c,b)))).


/* Rule 68 */

xkb_firealarm(
    [' Fire Alarm System is required ']) :-
                     xkb_identify(' Group F Division 1 '),

parmset(occupant_load,c,[c,d,e,f,g,h,j,k,l,m]).


/* Rule 69 */
```

```
xkb_firealarm(
    [' Fire Alarm System is required ']) :-
                    xkb_identify(' Group F Division 2 '),
                    parmset(occupant_load,c,[e,g,h,j,k,l,m]).



/* Rule 70 */

xkb_firealarm(
    [' Fire Alarm System is required ']) :-
                    xkb_identify(' Group F Division 3 '),
                    parmset(occupant_load,c,[m]).


/* Rule 71 */

xkb_firealarm(
    [' Fire Alarm System is not required ']) :-
                   \+ xkb_firealarm(
                     [' Fire Alarm System is required ']).


xkb_question(hotel_or_motel,
   [' Is the building used as a hotel or motel ']).

xkb_question(exit,
 [' Is the public corridor or exit shared by less than ',
  ' 4 dwelling units ?']).

xkb_question(access,
  [' Does each dwelling unit have direct access to the
     outdoors by a door at ground level or or a door
     providing access by a balcony with an exit stair to
     ground level ?']).

xkb_question(suite_access,
  [' Does each suite or sleeping room not within a suite
     have direct access to the outdoors by a door at ground
     level or door providing direct access by a balcony with
     an exit stair to ground  level ?']).


end_of_file.
```

```
/* Safety.pl  */

/* Safety Requirements within floor areas  */
:- dynamic xkb_safety/1.
:- retractall(xkb_safety(_)).

xkb_identify(' Group A ') :-
                  ( xkb_identify(' Group A Division 1 ')
                  | xkb_identify(' Group A Division 2 ')
                  | xkb_identify(' Group A Division 3 ')
                  | xkb_identify(' Group A Division 4 ')).


xkb_identify(' Group B ') :-
                     (xkb_identify(' Group B Division 1 ')
                    | xkb_identify(' Group B Division 2 ')).


/* Rule 72 */

xkb_safety(
    [' Every room and every suite shall have 2 egress
       doorways placed in such a manner that one doorway
       could provide egress from the room if the other
       doorway becomes inacessible to the occupants due to a
    '   fire which might originate in the room or suite.'])
              :-
                      ( prop(occupant_load_room)
                     | (xkb_identify(' Group A '),
                        (parmset(area_room,c,[d,e])
                                                          |
parmset(distance_egress_door,c,[c,d])))
                     | ( xkb_identify(' Group B '),
                        (parmset(area_room,c,[b,c,d,e])
                                                          |
parmset(distance_egress_door,c,[b,c,d])))
                     | (( xkb_identify(' Group C '),
                       (\+ prop(dwelling_unit))),
                   (parmset(area_room,c,[c,d,e])
                                                          |
parmset(distance_egress_door,c,[c,d])))

                     | (xkb_identify(' Group D '),
                        (parmset(area_room,c,[e])
                                                          |
parmset(distance_egress_door,c,[d])))

                     |  ( xkb_identify(' Group E '),
                        (parmset(area_room,c,[d,e])
                                                          |
parmset(distance_egress_door,c,[c,d])))
```

```
                    |   xkb_identify(' Group F Division 1 ')

                   |  ( xkb_identify(' Group F Division 2 '),
                        (parmset(area_room,c,[d,e])
                                                          |
parmset(distance_egress_door,c,[b,c,d])))

                     | (xkb_identify(' Group F Division 3 '),
                        (parmset(area_room,c,[e])
                                                          |
parmset(distance_egress_door,c,[c,d]))))).


xkb_safety(
  [' Corridors shall',
   ' (a) be separated from the remainder of the building by
         a fire separation having a fire-resistance rating
        of not less than 1 h, except that ',
 '(i) the fire-resistance rating need not be greater than
      3/4 h where the floor assembly is permitted to have a
      3/4 h fire resistance rating, and',
   ' (ii) no fire-resistance rating is required if the floor
         area is sprinklered,',
   ' (b) be equipped with emergency lighting to a level of
        at least 50 lux at floor level,',
   ' (c) have surface finishes with a flame-spread rating of
        not more 150.']) :-
            xkb_identify(' Group A ').


xkb_safety(
  [' Every door that opens onto a corridor or other facility
     that provides access to exit from a suite shall swing  on
a vertical axis, and the door shall swing in the
direction of exit travel.']) :-
                prop(occupant_load_room).

xkb_safety(
  ['  The  seating  area  shall  be  separated  from  adjacent
     occupancies by a fire separation having a
     fire-resistance rating of at least 1 h, except that a
     fire separation having a fire-resistance rating of
     3/4 h may be used where the floor assembly is not',
   ' required to have a fire-resistance rating greater than
     3/4 h.']) :-
                    xkb_identify(' Group A Division 1 '),
                      prop(occupant_load_seating_area).

xkb_safety(
  [' Each balcony shall have at least 3 separate exits.'])
        :-
```

```
            xkb_identify(' Group A Division 4 '),
            parm(occupant_load_balcony,c,a).

xkb_safety(
   [' Each balcony shall have at least 4 separate exits.'])
         :-
            xkb_identify(' Group A Division 4 '),
            parm(occupant_load_balcony,c,b).

xkb_safety(
  [' Every seat shall be located so that the travel distance
    does not exceed 45 m measured along the path of travel
    from the seat to',
   ' (a) the ground',
   ' (b) an exit',
   ' (c) an opening to a passageway leading from the seating
         area,or',
   ' (d) an opening through the seating deck structure such
         as a portal',
   ' ',
   ' The capacity of means of egress shall be based on  ',
   ' (a) 1 unit of exit width for each 300 persons for',
   '       (i) aisles,',
   '       (ii) stairs other than exit stairs, and',
   '        (iii) ramps and level passageways in vomitories
                 and in exits,',
   ' (b)  1 unit of width for each 225 persons for exit
         stairs.',
   ' ',
   ' Aisles',
   ' (a) shall be located so that there are not more than 20
         seats between any seat and the nearest aisle',
   ' (b) shall be at least 1200 mm in width,except that an
         aisle serving fewer than 60 persons may be 750 mm
         in width, and',
   ' (c) shall not have steps unless the gradient of the
         aisle exceeds 1 in 8.']) :-
            xkb_identify(' Group A Division 4 ').


xkb_safety(
   [' (a) The book storage room shall be separated from the
         remainder of the building by a 2 h fire
         separation, or ',
    ' (b) the building shall be sprinklered.']) :-
                  xkb_identify(' Group A Division 2 '),
                  prop(library),
                  (prop(area_book_storage)
                   | prop(height_book_stack)).

xkb_safety(
   [' (1) Any portion of a building in which 3 or more
```

```
        bowling lanes are located shall be separated from
        other occupancies by at least a 1 h fire
        separation',
    ' (2) Subsidiary occupancies such as offices,cocktail
        lounges and lunch counters operated in connection
        with 3 or more bowling lanes shall be separated by
        at least a 1 h fire separation where the combined
        area of these subsidiary occupancies exceeds
            150 m2.']) :-
                xkb_identify(' Group A Division 2 '),
                    prop(bowling_alley).

xkb_safety(
  [' (1) Structural members supporting the floor of any
        stage for theatrical performances shall be of
        noncombustible construction unless the building is
        permitted to be of combustible construction.',
   ' (2) Stages for thetrical performances and ancillary
        spaces,such as workshops,dressing rooms and storage
        areas,shall be sprinklered.',
   ' (3) A 1 h fire separation shall be provided between
        every stage for theatrical performances and
        ancillary spaces, such as workshops,',
   '     dressing rooms and storage areas.',
   ' (4) Every stage for theatrical performances and
        ancillary spaces shall be separated from the
        audience space by a fire separation having a
        fire-resistance rating of at least 1 h,except
        for a proscenium opening which shall be protected
        with',
   ' (a) an unframed fire curtain when the opening does not
        exceed 20 m in width,or',
   ' (b) a semi-rigid fire curtain when the opening is more
        than 20 m in width.']) :-
                xkb_identify(' Group A Divis: n 1 ').


xkb_safety(
 [' (1) Sleeping rooms and patients bedrooms shall be
        separated from adjacent rooms by a fire separation
 having a fire-resistance rating of at least 1 h,
 except that the fire-resistance rating need not be
   greater than 3/4 h where the floor assembly is not   '
  required to exceed 3/4 h.',
  ' (2) Corridors used by the public or serving patients
        bedrooms shall',
  ' (a) be separated from adjacent rooms by a fire
        separation  having a fire- resistance rating
        conforming to section (1).',
  ' (b) be equipped with emergency lighting to a level of at
        least 50 lux',
  ' (c) have surface finishes with a fire-spread rating of
```

```
        not more than 150,and',
    ' (d) have no dead-end portions unless the area served by
        the dead-end has a second and separate egress
        doorway from each room or suite not leading into a
        dead-end corridor.']) :-
                        xkb_identify(' Group B ').

xkb_safety(
  [' (1) Suites shall be separated from each other and the
        remainder of the building by a fire separation
        having a fire-resistance rating of at least 1h,
        except that a 3/4 h fire-resistance rating is
        permitted where the fire-resistance rating of the
        floor assembly is not required to exceed 3/4h.',
    ' (2) Floors that separate storeys within a dwelling unit
        need not be constructed as fire separations provided
        the distance between the lowest floor level and the
        uppermost floor level does not exceed 6 m',
    '     and provided that each dwelling unit is separated
        from the remainder of the building by',
    ' (a) a fire separation having a fire-resistance rating of
        at least 1 h where the building is',
    '     (i) 3 storeys or less in building height,or',
    '     (ii) sprinklered,and',
    ' (b) a fire separation having a fire-resistance rating of
        at least 2 h where the building is not sprinklered
        and is greater than 3 storeys in building height.'])
                    :-
                        xkb_identify(' Group C ').


xkb_safety(
  [' (1) Partitions of combustible construction may be used
        within floor areas,provided the partitions are not
         required to act as  fire separations',
    ' (a) where the floor area is sprinklered,or',
    ' (b) within spaces having an area not more than 500 m2
        where such spaces are separated from the remainder
        of the floor area by at least a 1 h fire separation
        of noncombustible construction',
    ' (2) Partitions of combustible construction shall be
        limited to',
    ' (a) wood studs covered on both sides by noncombustible
        cladding or gypsum wallboard.',
    ' (b) wood studs covered on both sides by fire-retardant
        treated wood having a flame-spread rating of not
        more than 25,',
    ' (c) glass in wood sash,or',
    ' (d) solid lumber not less than 38 mm thickness',
    ' (3) Interior finish material used on the wall or
        ceiling of every room shall have a flame-spread
```

```
          rating of not  more than 150 .']) :-
                  (xkb_identify(' Group D ')
                   | xkb_identify(' Group E ')).

xkb_identify(' Group F ') :-
                (xkb_identify(' Group F Division 1 ')
                | xkb_identify(' Group F Division 2 ')
                | xkb_identify(' Group F Division 3 ')).


xkb_safety(
   [' (1) Basements or cellars shall not be used for the
          storage, manufacture or handling of volatile
          solids,liquids or gases that generate explosive
          air-vapour mixtures or for processes that involve
          explosive dusts.',
    ' (2) Interior finish material used on the wall or
          ceiling of every room shall have a flame-spread
          rating of not more than  150.']) :-
                              xkb_identify(' Group F ').



xkb_safety(
   [' Every process room where hazardous substances are used
      or intended to be used shall be separated from the
      remainder of the building by a 2 h fire separation
      unless the room is protected by a suitable fire
      extinguishing system.']) :-
                  ( xkb_identify(' Group F Division 2 ')
                   | xkb_identify(' Group F Division 3 ')).

xkb_safety(
   [' (1) The clear height of every storey shall be at least
          2 m.',
    ' (2) A storage garage shall be separated from other
   occupancies  by  at  least  1.5  h  fire  separation.'])
               :-
                        xkb_identify(' Group F Division 3 '),
                             prop(storage_garage).



xkb_safety(
  [' (1) A repair garage shall be separated from other
         occupancies by at least 2 h fire separation.']) :-
                  xkb_identify(' Group F Division 2 '),
                  prop(repair_garage).
```

```
xkb_question(occupant_load_room,
   [' Is the occupant load for the room greater than 60
      persons ?']).



xkb_question(area_room,
   [' What is the maximum area of the room or suite in
      square metres ?',
    ' choose the closest number greater than the maximum
      area of the room',
    ' ',
    ' (a) 75   (b) 100   (c)  150  (d) 200',
    ' (e) greater than 200 ']).


xkb_question(distance_egress_door,
   [' What is the maximum distance to the egress door in
      metres ?',
    ' choose the closest number greater than the maximum
distance to the egress door',
    ' ',
    ' (a) 10   (b) 15   (c) 25   (d) greater than 25 ']).


xkb_question(occupant_load_seating_area,
   [' Does the seating area contain more than 200 persons
        ?']).

xkb_question(occupant_load_balcony,
   [' How many persons does the balcony contain ? ',
    ' (a) more than 1000 persons',
    ' (b) more than 4000 persons ']).


xkb_question(dwelling_unit,
   [' Is the building used as a dwelling unit ?']).

xkb_question(library,
   [' Is the building used as a library ?']).

xkb_question(area_book_storage,
   [' Does a book storage room in the library exceed 250 m2
      in area ?']).

xkb_question(height_book_stack,
   [' Do the book stacks exceed 10 m in height or ',
    ' penetrate more than 1 storey ?']).

xkb_question(bowling_alley,
```

```
   [' Is the building used as a bowling alley ?']).

xkb_question(repair_garage,
   [' Is the building used as a repair garage ?']).


end_of_file.
```

```
/* Exit.pl  */

/* Requirements for exits  */

/* Rule 92  */

xkb_exit(
  [' A floor area may be served by 1 exit.']) :-
                      parmset(number_storeys,c,[a,b]),
                      \+ prop(occupant_load_room),
                     (( xkb_identify(' Group A '),
                       (parmset(area_room,c,[d,e])
                        | parmset(travel_distance,c,[c,d])))

                       | (xkb_identify(' Group B '),
                          (parmset(area_room,c,[b,c,d,e])
                                                          |
parmset(travel_distance,c,[b,c,d])))

                         | (xkb_identify(' Group C '),
                            (parmset(area_room,c,[c,d,e])
                             | parmset(travel_distance,c,[c,d])))

                         | (xkb_identify(' Group D '),
                            (parmset(area_room,c,[e])
                             | parmset(travel_distance,c,[d])))

                         | (xkb_identify(' Group E '),
                            (parmset(area_room,c,[d,e])
                             | parmset(travel_distance,c,[c,d])))

                    | (xkb_identify(' Group F Division 2 '),
                            (parmset(area_room,c,[d,e])
                                                          |
parmset(travel_distance,c,[b,c,d])))

                   | (xkb_identify(' Group F Division 3 '),
                            (parmset(area_room,c,[e])
                                                          |
parmset(travel_distance,c,[c,d])))).


/* Rule 93  */

xkb_exit(
  [' Every floor area shall be served by not fewer than 2
     exits.']) :-
                        (\+ xkb_exit(
                 [' A floor area may be served by 1 exit.'])).


/* Rule 94  */
```

```
xkb_exit(
  [' (1) The least distance between 2 required exits from a
       floor area shall be',
   ' (a) one half the maximum diagonal dimension of the
       floor area,but need not be more than 9 m for a
       floor area having a floor area having a public
       corridor; or',
   ' (b) one half the maximum diagonal dimension of the
       floor area, but not less than 9 m for all other
       floor areas.',
   ' (2) The minimum distance between exits shall be the
  shortest distance that smoke would have to travel
  between the required exits, assuming that the smoke
  will not penetrate  an intervening fire
  separation.']).


/* Rule 95  */

xkb_exit(
  [' Where more than one exit is required from a floor
     area,such exits shall be located so that the travel
     distance to at least 1 exit shall be not more than 25 m
          .']) :-
                    xkb_identify(' Group F Division 1 ').


/* Rule 96  */

xkb_exit(
  [' Where more than one exit is required from a floor
     area,such exits shall be located so that the travel
     distance to at least 1 exit shall be not more than 45 m
          .']) :-
              \+ xkb_identify(' Group F Division 1 '),
                   \+ prop(unsprinklered).


/* Rule 97  */

xkb_exit(
  [' Where more than one exit is required from a floor
     area, such exits shall be located so that the travel
     distance to at least 1 exit shall be not more than 40 m
          .']) :-
                    xkb_identify(' Group D ').


/* Rule 98  */

xkb_exit(
  [' Where more than one exit is required from a floor
```

```
     area,such exits shall be located so that the travel
     distance to at least 1 exit shall be not more than 30 m
          .']) :-
                \+ xkb_identify(' Group F Division 1 '),
                     \+ xkb_identify(' Group D ').


/* Rule 99  */

xkb_exit(
  [' (1) The clear width of any corridor used as an exit
         shall be at least 1100 mm.',
   ' (2) The clear width of any exit serving patients in bed
         shall be at least 1100 mm.',
   ' (3) The width of an exit stair shall be at least 1100 mm
         where the stair serves more than 3 storeys above
         grade or more than 1 storey below grade.',
   ' (4) Every exit shall have a headroom clearance of at
         least 2150 mm.',
   ' ',
   ' (5) Flame - spread rating for exits',
   '     ------------------------------',
   ' (a) The flame-spread rating of a wall or ceiling in an
         exit shall not exceed 25.',
   ' (b) The flame-spread rating of interior finish for
         doors,door frames and trim in exits may exceed 25
         provided such finish has a flame-spread rating of
         not more than 150 and does not exceed 10 per cent of
         the wall or ceiling areas.']).


xkb_question(travel_distance,
   [' What is the maximum distance from any point in the
      floor area to an exit measured along the path of exit
      travel,in metres ?',
    ' ',
    ' choose the closest number greater than the maximum
      travel distance',
    ' (a) 10    (b) 15    (c) 25    (d) greater than 25 ']).


end_of_file.
```

```
/* Plumbing.pl  */

/* Health Requirements  */

/* Plumbing Facilities  */

/*  Rule 100  */

xkb_plumbing(
   [' Minimum number of water closets = 1 for males ',
    ' Minimum number of water closets = 1 for females ']) :-
                    (xkb_identify(' Group A Division 1 ')
                    | xkb_identify(' Group A Division 3 ')
                    | xkb_identify(' Group A Division 4 ')
                    | parmset(assembly_use,c,[b,c,s,t])),
                    parm(number_persons_each_sex,c,a).


/*  Rule 101   */

xkb_plumbing(
   [' Minimum number of water closets = 1 for males ',
    ' Minimum number of water closets = 2 for females ']) :-
                    (xkb_identify(' Group A Division 1 ')
                    | xkb_identify(' Group A Division 3 ')
                    | xkb_identify(' Group A Division 4 ')
                    | parmset(assembly_use,c,[b,c,s,t])),
                    parm(number_persons_each_sex,c,b).


/*  Rule 102   */

xkb_plumbing(
   [' Minimum number of water closets = 2 for males ',
    ' Minimum number of water closets = 3 for females ']) :-
                    (xkb_identify(' Group A Division 1 ')
                    | xkb_identify(' Group A Division 3 ')
                    | xkb_identify(' Group A Division 4 ')
                    | parmset(assembly_use,c,[b,c,s,t])),
                    parm(number_persons_each_sex,c,c).


/*  Rule 103   */

xkb_plumbing(
   [' Minimum number of water closets = 2 for males ',
    ' Minimum number of water closets = 4 for females ']) :-
                    (xkb_identify(' Group A Division 1 ')
                    | xkb_identify(' Group A Division 3 ')
                    | xkb_identify(' Group A Division 4 ')
                    | parmset(assembly_use,c,[b,c,s,t])),
```

```
                    parm(number_persons_each_sex,c,d).


/*  Rule  104  */

xkb_plumbing(
   [' Minimum number of water closets = 3 for males ',
    ' Minimum number of water closets = 5 for females ']) :-
                    (xkb_identify(' Group A Division 1 ')
                    | xkb_identify(' Group A Division 3 ')
                    | xkb_identify(' Group A Division 4 ')
                    | parmset(assembly_use,c,[b,c,s,t])),
                    parm(number_persons_each_sex,c,e).


/*  Rule 105   */

xkb_plumbing(
   [' Minimum number of water closets = 4 for males ',
    ' Minimum number of water closets = 6 for females ']) :-
                    (xkb_identify(' Group A Division 1 ')
                    | xkb_identify(' Group A Division 3 ')
                    | xkb_identify(' Group A Division 4 ')
                    | parmset(assembly_use,c,[b,c,s,t])),
                    parm(number_persons_each_sex,c,f).


/*  Rule  106  */

xkb_plumbing(
   [' Minimum number of water closets = 5 for males ',
    ' Minimum number of water closets = 7 for females ']) :-
                    (xkb_identify(' Group A Division 1 ')
                    | xkb_identify(' Group A Division 3 ')
                    | xkb_identify(' Group A Division 4 ')
                    | parmset(assembly_use,c,[b,c,s,t])),
                    parm(number_persons_each_sex,c,g).


/*  Rule  107  */

xkb_plumbing(
   [' Minimum number of water closets = 6 for males ',
    ' Minimum number of water closets = 8 for females ']) :-
                    (xkb_identify(' Group A Division 1 ')
                    | xkb_identify(' Group A Division 3 ')
                    | xkb_identify(' Group A Division 4 ')
                    | parmset(assembly_use,c,[b,c,s,t])),
                    parm(number_persons_each_sex,c,h).
```

```
/*  Rule  108  */

xkb_plumbing(
[' Minimum number of water closets       number of males -
                                                    400 ',
 '            for  males          = 7 +----------------------
-',                                                  200



 ' ',
 ' ',
 ' Minimum number of water closets      number of females -

                  for females      = 8+                    400
                                      ---------------------',
                                                  150      '])
                    :-
                     (xkb_identify(' Group A Division 1 ')
                     | xkb_identify(' Group A Division 3 ')
                     | xkb_identify(' Group A Division 4 ')
                     | parmset(assembly_use,c,[b,c,s,t])),
                     parm(number_persons_each_sex,c,j).


/* Rule  109  */

xkb_plumbing(
 [' Minimum number of water closets for each sex = 1 ']) :-
                      (xkb_identify(' Group D ')
                      | parmset(assembly_use,c,[e,h,i,k,q])),
                       parm(number1_persons_each_sex,c,a).



/* Rule  110  */

xkb_plumbing(
 [' Minimum number of water closets for each sex = 2 ']) :-
                      (xkb_identify(' Group D ')
                      | parmset(assembly_use,c,[e,h,i,k,q])),
                       parm(number1_persons_each_sex,c,b).



/* Rule  111  */

xkb_plumbing(
['Minimum number of water closets    number of persons of
```

```
                                        each sex  - 50 ',
'                for each sex   = 3  + --------------------

                                                50          ',
' ',
'                          ')] :-
                      (xkb_identify(' Group D ')
                      | parmset(assembly_use,c,[e,h,i,k,q])),
                       parm(number1_persons_each_sex,c,c).


/* Rule  112  */

xkb_plumbing(
 [' The number of water closets shall be at least one
    fixture for each 30 males and one fixture for each 25
    females.']) :-
                      parmset(assembly_use,c,[p,v]).



/* Rule  113  */

xkb_plumbing(
  [' The number of water closets shall be at least one
     fixture for each 150 persons of each sex.']) :-
                      parmset(assembly_use,c,[g,u]).


/* Rule  114  */

xkb_plumbing(
  [' The number of water closets shall be determined on the
     basis of the special needs of such occupancies .']) :-
                           xkb_identify(' Group B ').


/* Rule  115  */

xkb_plumbing(
  [' The number of water closets shall be at least one
     fixture for each 10 persons of each sex.']) :-
                       xkb_identify(' Group C ').


/* Rule  116   */

xkb_plumbing(
  [' The number of water closets shall be at least one
     fixture for each 300 males and one fixture for each 150
     females.']) :-
                          xkb_identify(' Group E ').
```

```
/* Rule  117  */

xkb_plumbing(
  [' Minimum number of water closets for each sex =  1 '])
               :-
                    xkb_identify(' Group F '),
                    parm(number2_persons_each_sex,c,a).


/* Rule  118  */

xkb_plumbing(
  [' Minimum number of water closets for each sex =  2 '])
               :-
                    xkb_identify(' Group F '),
                    parm(number2_persons_each_sex,c,b).


/* Rule  119  */

xkb_plumbing(
  [' Minimum number of water closets for each sex =  3 '])
          :-
                    xkb_identify(' Group F '),
                    parm(number2_persons_each_sex,c,c).


/* Rule  120  */

xkb_plumbing(
  [' Minimum number of water closets for each sex =  4 '])
               :-
                    xkb_identify(' Group F '),
                    parm(number2_persons_each_sex,c,d).


/* Rule  121  */

xkb_plumbing(
  [' Minimum number of water closets for each sex =  5 '])
          :-
                    xkb_identify(' Group F '),
                    parm(number2_persons_each_sex,c,e).


/* Rule  122  */

xkb_plumbing(
['Minimum number of water closets       number of persons of
                                             each sex  - 100
',
```

```
'              for each sex      = 6 +  ------------------
                                                  30     ',
' ',
'                                                 ']) :-
                    xkb_identify( ' Group F '),
                    parm(number2_persons_each_sex,c,f).


xkb_question(assembly_use,
  [' Is the building used as one of the following list ?',
   ' ',
   ' (a) Art galleries                    (b) Lecture halls',
   ' (c) Auditoria                        (d) Libraries',
   ' (e) Bowling alleys                       (f) Licensed
                                          beverage',
   ' (g) Churches and similar            establishments ',
   '     places of worship                (h) Lodge rooms',
   ' (i) Clubs,nonresidential             (j) Museums ',
      ' (k) Community halls                    (l) Passenger
                                           stations',
   ' (m) Court rooms                      and depots',
   ' (n) Dance halls                      (o) Recreational
                                           piers',
   ' (p) Day-care centres                 (q) Restaurants',
   ' (r) Exhibition halls                  (s) Schools and
                                           colleges',
   '                                      nonresidential',
   ' (t) Gymnasia                          (u) undertaking
                                           premises',
   ' (v) Primary schools                  (w) none of the
                                           above']).


xkb_question(number_persons_each_sex,
   [' What is the number of persons of each sex ?',
    ' ',
    ' (a) 1 - 10          (b)  26  -  50          (c)  51  -
                                                      75',
    ' (d) 76 - 100        (e)  101 - 150          (f) 151  -
                                                      200',
    ' (g) 201 - 300        (h)  301 - 400           (j) over
                                                    400  ']).


xkb_question(number1_persons_each_sex,
   [' What is the number of persons of each sex ?',
    ' ',
    ' (a) 1 - 25          (b)  26  -  50          (c) over 50
     ']).
```

```
xkb_question(number2_persons_each_sex,
   [' What is the number of persons of each sex ?',
    ' ',
    ' (a) 1 - 10          (b)  11  -  25          (c)  26  -
                                                      50',
    ' (d) 51 - 75         (e)  76  - 100           (f)  over
                                                     100']).

end_of_file.
```

```
/*  MENUSYST.PL */
/* A menu system written in Prolog  */

/* Requires procedure menu on file MENU.PL */
:- ensure_loaded('menu.pl').
:- ensure_loaded('building.pl').
:- ensure_loaded('query.pl').


/*
 * Menu navigation routines
 */

/* back(Path)
 *  Go to the previous menu. If none, write an
 *  error message and return to the current menu.
 */

back([Head|Tail]) :-  /* back to previous menu  */
    !,
    Goal =.. [Head,Tail],     /* note comma! */
    call(Goal).

back([]) :-
    Goal =.. [CurrentMenu,[]],
    write('Cannot back up further'),write(Goal),nl,
    call(Goal).

/* forward(NewMenu,CurrentMenu,Path)
 *    Go to NewMenu, passing it a correctly constructed path.
 */

forward(NewMenu,CurrentMenu,Path) :-  /* forward to another
menu */
    Goal =.. [NewMenu,[CurrentMenu|Path]],
    call(Goal).

/* application(Name,CurrentMenu,Path)
 *   Execute the specified application, then
 *   return to the current menu.
 */

application(Name,CurrentMenu,Path)   :-     /*   execute   an
application */
    call(Name),
    Goal =.. [CurrentMenu,Path],
    call(Goal).

/*
 *  Menus
 */
```

```
/*  Each menu takes an argument giving the list of menus
    through which it was reached (most recent first).
 */

menuA(_) :-
   nl,
   write('Main menu (menu A)'),nl,
   menu([item('Go to menu B', forward(menuB,menuA,[])),
         item('Go to menu C to choose the applications',
                                forward(menuC,menuA,[])),
         item('Exit to Unix',application(exit_to_unix,_,_)),
                          i t e m ( ' E x i t     t o
Prolog',application(exit_to_prolog,_,_))],
         Goal),
   call(Goal).

menuB(Path) :-
    nl,
    write('Menu B'),nl,
    menu([item('Go to menu D', forward(menuD,menuB,Path)),
          item('Go to menu E', forward(menuE,menuB,Path)),
          item('Previous menu',back(Path))],
          Goal),
    call(Goal).

menuC(Path) :-
    nl,
    write('Menu C'),nl,
    menu([item('Identify the building',

application(building_classification,menuC,Path)),
          item('Fire Regulation',
                  application(fire_regulation,menuC,Path)),
          item('Fire Alarm',
                  application(fire_alarm,menuC,Path)),
          item('Term Definition',
                  application(term_definition,menuC,Path)),
          item('Safety Requirements within floor areas',
                  application(safety,menuC,Path)),
          item('Requirements for exits',
                  application(exit_area,menuC,Path)),
          item('Plumbing facilities',
                  application(plumbing,menuC,Path)),
          item('Type a query,
                variable X = number of storeys,
                variable Y = building area',
                  application(type_a_query,menuC,Path)),
          item('Previous menu',back(Path))],
          Goal),
    call(Goal).
```

```
menuD(Path) :-
    nl,
    write('Menu D'),nl,
       menu([item('Exit to Prolog',
application(exit_to_prolog,_,_)),
          item('Jump to menu C',forward(menuC,menuD,Path)),
          item('Previous menu',back(Path))],
          Goal),
     call(Goal).

menuE(Path) :-
    nl,
    write('Menu E'),nl,
       menu([item('Exit to UNIX',
application(exit_to_unix,menuE,Path)),
          item('Previous menu', back(Path)),
                    item('Jump to main
menu',forward(menuA,menuE,Path))],
          Goal),
      call(Gcal).


/* The applications */

building_classification :- xshell,nl.
fire_regulation :- xshell1,nl.
fire_alarm :- xshell2,nl.
term_definition :- xshell3,nl.
safety :- xshell4,nl.
exit_area :- xshell5,nl.
plumbing :- xshell6,nl.
exit_to_unix :- halt.
exit_to_prolog :- abort.
type_a_query :- query,nl.

/* Starting query */
:- menuA(_).

end_of_file.
```

```
/* MENU.PL    */
/*
 * menu(Menu,Result)
 *  Displays a menu of up to nine items and returns the
 *  user's choice. A menu is a list of items,each of the
 *  form item(Message,Value) where Message(an atom, in
 *  single quotes) is the message to be displayed and Value
 *  is the value to be returned in Result if the user
 *  chooses this item.
 */


menu(Menu,Result) :- menu_display(Menu,49,Last),
                     menu_choose(Menu,49,Last,Result),
                     nl.

/* Display all the messages and simultaneously count them.
        The count starts at 49 (ASCII code for '1').  */

menu_display([],SoFar,Last) :- !,
                    /* not an item , so don't use this number
                       */
                        Last is SoFar-1.

menu_display([item(Message,_)|Tail],SoFar,Last) :-
     put(32),
     put(32),
     put(SoFar),                  /* appropriate digit  */
     put(32),                     /* blank  */
     write(Message),
     nl,
     Next is SoFar+1,
     menu_display(Tail,Next,Last).

   /* Get the user's choice. If invalid, make him try again.
     */

menu_choose(Menu,First,Last,Result) :-
    write('Your choice ('),
    put(First),
    write(' to '),
    put(Last),
    write('): '),
    get(Char),
    menu_choose_aux(Menu,First,Last,Result,Char).


menu_choose_aux(Menu,First,Last,Result,Char) :-
    Char>=First,
    Char=<Last,
    !,
    menu_select(Menu,First,Char,Result).
```

```
menu_choose_aux(Menu,First,Last,Result,Char) :-
    put(7),        /* beep  */
    put(13),       /* carriage return  */
  menu_choose(Menu,First,Last,Result).

  /* Find the appropriate item to return for Char  */

menu_select([item(_,Result)|_],First,First,Result) :- !.

menu_select([Head|Tail],First,Char,Result) :-
     NewFirst is First+1,
     menu_select(Tail,NewFirst,Char,Result).

end_of_file.
```

```
/* READNUM.PL */
/* Input procedure for integers and floating-point      numbers
*/

/* Requires procedure readstring defined in file  READSTR.PL
*/
:- ( clause(readstring(_),_) ; consult('READSTR.PL') ).

/*****************************************************************
****
 * readnumber(Result)                                            *
 * Accepts a string from the user and interprets it as a         *
 * number (integer or floating point). Negative numbers          *
are  accepted; E format is not.                                  *

*****************************************************************
***/

readnumber(Result) :- readstring(S),
                      readnumber_start(S,Result).

/* readnumber_start(String,Result)
 *    checks for initial blanks or minus sign, then              *
   passes control to readnumber_aux with state                   *
variables initialized.
 */

readnumber_start([32|Tail],Result) :-
                                    /*    discard    leading
                                        blanks  */
                    !,
                    readnumber_start(Tail,Result).

readnumber_start([45|Tail],Result) :-
                                       /*    begins    with
                                        minus sign */
                    !,
                    readnumber_aux(Tail,no,-1,0,Result).

readnumber_start(String,Result) :-
                                  /* does not begin
                                with minus*/
                    readnumber_aux(String,no,1,0,Result).

/* readnumber_aux(String,Point,Divisor,SoFar,Result)
 *   works through String one character at a time.
 *  Point is 'yes' if the point has been found, 'no' if          *
 not.
 *  Divisor is 1 until the point is encountered,after
 *  which it becomes 10,100,etc., in succession.
 *  If number is negative,divisor is -1, -10, etc.
 *  Sofar represents the part of the number already read.
 */
```

```
readnumber_aux([Digit|Tail],no,Divisor,SoFar,Result) :-
        readnumber_value(Digit,Value),
         !,          /* a digit to the left of the point */
        NewSoFar is SoFar*10+Value,
        readnumber_aux(Tail,no,Divisor,NewSoFar,Result).

readnumber_aux([46|Tail],no,Divisor,SoFar,Result) :-
         !,       /* the decimal point itself */
        NewSoFar is SoFar*10+Value,
        NewDivisor is Divisor*10,
        readnumber_aux(Tail,yes,NewDivisor,NewSoFar,Result).

readnumber_aux([],_,1,Result,Result) :- !.
                                                    /*    all
                                done,positiveinteger */

 readnumber_aux([],_,-1,SoFar,Result) :- !,
                                /*    all    done,    negative
                                     integer  */
                                 Result is -(SoFar).

readnumber_aux([],yes,Divisor,SoFar,Result) :-
               !,                /* all done, it's floating
                           point  */
               Result is SoFar/Divisor.

readnumber_aux(_,_,_,_,Result) :- /* unrecognized character */
                write('Number    expected.    Try
                                     again:'),
            readnumber(Result).

/* readnumber_value(ASCII,Number)
 *  converts ASCII codes of digits to numeric values.
 */

readnumber_value(48,0).
readnumber_value(49,1).
readnumber_value(50,2).
readnumber_value(51,3).
readnumber_value(52,4).
readnumber_value(53,5).
readnumber_value(54,6).
readnumber_value(55,7).
readnumber_value(56,8).
readnumber_value(57,9).

end-of-file.
```

```
/* READSTR.PL    */
/* Contains utility procedures readstring, writestring,
and readatom  */

/*************************************************************
****
 * readstring(Result)                                          *
 *    Displays a one-character prompt, then accepts a line
*    of input from the keyboard and returns it as a          *
  string(list of ASCII codes).                                *
The user can use Backspace (Ctrl-H) to make                   *
corrections,and is prevented from backspacing past    *   the
beginning of the line.                                  *

*************************************************************
***/

readstring(Result) :- put(62),       /* Prompt character is
                                        '>'  */
                      readstring_start(0,Result).

readstring_start(Count,X) :- get0(Char),
                             readstring_aux(Count,Char,X).

readstring_aux(Count,13,[]) :- !.          /* Return  */
readstring_aux(Count,10,[]) :- !.          /* New line */


readstring_aux(0,8,Result)    :-    !,/*    Backspace    past
                                    beginning */
                             readstring(Result).  /* start
                                                    */

readstring_aux(Count,8,Result) :- !, /* Backspace */
                          put(32),  /*  Print  a  blank,then
                                    */
                          put(8), /* backspace again */
                          NewCount is Count-1,
                          readstring_start(Newcount,Tail),
                          readstring_cons(8,Tail,Result).

readstring_aux(Count,Char,Result) :- /* Normal character
                                    */
                              NewCount is Count+1,

readstring_start(NewCount,Tail),

readstring_cons(Char,Tail,Result).

readstring_cons(Char,[8|Tail],Tail) :- Char\==8, !.
readstring_cons(Char,Tail,[Char|Tail]).
```

```
/*****************************************************************
****
 * writestring(String)                                           *
 * Displays a string (a list of ASCII codes).                    *

*****************************************************************
***/

writestring([]).
writestring([Head|Tail]) :- put(Head),writestring(Tail).

/*****************************************************************
****
 * readatom(Atom)                                                *
 *  Reads a string and then converts it to an atom.              *

*****************************************************************
***/
readatom(Atom) :- readstring(String), name(atom,String).

end-of-file.
```

```
/* WRITELN.PL     */

/* If the argument is a list, display the elements        */
/* one by one and start a new line after each.            */
/* Otherwise display the argument and start a new line.   */

writeln([]) :- !.
writeln([Head|Tail]):- !,write(Head),nl,writeln(Tail).
writeln(Arg) :- write(Arg),nl.
end_of_file.
```

```
/* YES.PL     */

/*************************************************************
 * yes(Question)                                             *
 * Displays Question,insists that the user type              *
 * Y or N (upper or lower case), and then succeeds           *
 * if user typed Y , or fails if user typed N.               *
 *************************************************************/

yes(Question) :- write(Question),
                 get(Char),
                 yes_aux(Char).

yes_aux(89)   :- !.          /* Y */

yes_aux(121)  :- !.          /* y */

yes_aux(78)   :- !,fail.     /* N */

yes_aux(110)  :- !,fail.     /* n */

yes_aux(_)    :- put(7),     /* beep */
                 write(' [Type Y or N]:'),
                 get(Char),
                 yes_aux(Char).

end_of_file.
```

```
                    nl,nl,write('safety requirements within
                                floor areas '),
                    finda(Tempa),
                    nl,
                    safety_requirement(Tempa,ID1),
                    writeln(ID1),
                    retractall(finda(Tempa)).


write_exit :-
                    nl,nl,write('requirements for exits '),
                    finda(Tempa),
                    nl,
                    exit_requirement(Tempa,ID1),
                    writeln(ID1),
                    retractall(finda(Tempa)).


write_plumbing :-
                    nl,nl,write('health requirements for
                                 plumbing facilities '),
                    finda(Tempa),
                    nl,
                    plumbing_requirement(Tempa,ID1),
                    writeln(ID1),
                    retractall(finda(Tempa)).

end_of_file.
```

```
/* query.pl */

/* A customized user interface for Prolog */
/*
 * query
 *   Get a query and display its solutions.
 */

:- ensure_loaded('firealarm_log.pl').
:- ensure_loaded('safety_log.pl').
:- ensure_loaded('exit_log.pl').
:- ensure_loaded('plumbing_log.pl').


query :-
            retractall(known(_,_)),
            retractall(find(_)),nl,
            retractall(finda(_)),nl,
            retractall(findl(_)),
            retractall(findq(_)),
            retractall(findr(_)),
            retractall(findrr(_)),
            retractall(findx(_)),
            retractall(findxx(_)),
            retractall(findy(_)),
            retractall(findyy(_)),
            writeln('Query '),
            writeln('Rules are defined as rule(A,S,X,Y,Z)'),
            writeln(' Use the following variables'),
            writeln(' X = number of storeys'),
            writeln(' Y = building area in m2'),
            writeln('  Z  =  number  of  streets  the  building
faces'),
            writeln(' A = rule number'),
            writeln('  S  =  status  whether  the  building  is
sprinklered or not'),
            writeln(' S = 1 (unsprinklered)'),
            writeln(' S = 2 (sprinklered)'),
            read(Q),
            asserta(find(Q)),
            (query1 | query2).

/*
 * Query has the format bagof(X,rule(A,S,X,Y,Z),L)
 */

query1 :-
            find(Q),
            arg(2,Q,R),
            assert(findr(R)),
            arg(3,R,X),
            Tempx = X,
```

```
            assert(findx(Tempx)),
            arg(4,R,Y),
            Tempy = Y,
            assert(findy(Tempy)),
            find_solution_1(Q).


/*
 * Query has the format rule(A,S,X,Y,Z)
 */

query2 :-

            find(Q),
            arg(2,Q,S),
            Temps = S,
            assert(finds(Temps)),
            arg(3,Q,X),
            Tempx = X,
            assert(findx(Tempx)),
            arg(4,Q,Y),
            Tempy = Y,
            assert(findy(Tempy)),
            arg(5,Q,Z),
            Tempz = Z,
            assert(findz(Tempz)),
            find_solution_2(Q).


query3 :-
            retractall(known(_,_)),
            retractall(find(_)),nl,
            retractall(finda(_)),nl,
            retractall(findl(_)),
            retractall(findq(_)),
            retractall(findr(_)),
            retractall(findrr(_)),
            retractall(findx(_)),
            retractall(findxx(_)),
            retractall(findy(_)),
            retractall(findyy(_)),
            writeln('Query : Fire Alarm '),
            writeln('Rules are defined as alarm(A,X,Y)'),
            writeln(' Use the following variables'),
            writeln(' X = number of storeys'),
            writeln(' Y = occupant load'),
            writeln(' A = rule number'),
            read(Q),
            asserta(find(Q)),
            find(Q),
            arg(2,Q,X),
```

```
            Tempx = X,
            assert(findx(Tempx)),
            arg(3,Q,Y),
            Tempy = Y,
            assert(findy(Tempy)),
            find_solution_3(Q).


query4 :-
            retractall(known(_,_)),
            retractall(find(_)),nl,
            retractall(finda(_)),nl,
            retractall(findl(_)),
            retractall(findq(_)),
            retractall(findr(_)),
            retractall(findrr(_)),
            retractall(findx(_)),
            retractall(findxx(_)),
            retractall(findy(_)),
            retractall(findyy(_)),
            retractall(findz(_)),
            retractall(findzz(_)),
            writeln('Query : safety requirements within floor
                     areas '),
            writeln('Rules are defined as safety(A,X,Y,Z)'),
            writeln(' Use the following variables'),
            writeln(' X = area of room or suite in m2'),
            writeln(' Y = distance to egress doorway in
                    meters'),
            writeln(' Z = occupant load'),
            writeln(' A = rule number'),
            read(Q),
            asserta(find(Q)),
            find(Q),
            arg(2,Q,X),
            Tempx = X,
            assert(findx(Tempx)),
            arg(3,Q,Y),
            Tempy = Y,
            assert(findy(Tempy)),
            arg(4,Q,Z),
            Tempz = Z,
            assert(findz(Tempz)),
            find_solution_4(Q).


query5 :-
            retractall(known(_,_)),
            retractall(find(_)),nl,
```

```
                retractall(finda(_)),nl,
                retractall(findl(_)),
                retractall(findq(_)),
                retractall(findr(_)),
                retractall(findrr(_)),
                retractall(findx(_)),
                retractall(findxx(_)),
                retractall(findy(_)),
                retractall(findyy(_)),
                retractall(findz(_)),
                retractall(findzz(_)),
                writeln('Query :  requirements for exits '),
                writeln('Rules are defined as exit(A,X,Y,Z)'),
                writeln(' Use the following variables'),
                writeln(' X = number of storeys'),
                writeln(' Y = area of room in m2'),
                writeln(' Z = travel distance in meters'),
                writeln(' A = rule number'),
                read(Q),
                asserta(find(Q)),
                find(Q),
                arg(2,Q,X),
                Tempx = X,
                assert(findx(Tempx)),
                arg(3,Q,Y),
                Tempy = Y,
                assert(findy(Tempy)),
                arg(4,Q,Z),
                Tempz = Z,
                assert(findz(Tempz)),
                find_solution_5(Q).


query6 :-
                retractall(known(_,_)),
                retractall(find(_)),nl,
                retractall(finda(_)),nl,
                retractall(findl(_)),
                retractall(findq(_)),
                retractall(findr(_)),
                retractall(findrr(_)),
                retractall(findx(_)),
                retractall(findxx(_)),
                retractall(findy(_)),
                retractall(findyy(_)),
                writeln('Query : Plumbing facilities '),
                writeln('Rules are defined as plumbing(A,X,Y)'),
                writeln(' Use the following variables'),
                writeln(' X = number of males'),
                writeln(' Y = number of females'),
                writeln(' A = rule number'),
                read(Q),
```

```
            asserta(find(Q)),
            find(Q),
            arg(2,Q,X),
            Tempx = X,
            assert(findx(Tempx)),
            arg(3,Q,Y),
            Tempy = Y,
            assert(findy(Tempy)),
            find_solution_6(Q).


/*
 * find_solutions(Q)
 *  call Q, then display Q with instantiations,
 *  then ask if more solutions are wanted.
 *
 */

find_solution_1(Q) :-
                      find(Q),
                      call(Q),
                      assert(findq(Q)),
                      write('Solution found:    '),
                      writeln(Q),
                      nl,
                      findq(Q),
                      arg(3,Q,L1),
                      LL =.. L1,
                      Temp1 = LL,
                      assert(findl(Temp1)),
                      findq(Q),
                      arg(2,Q,RR),
                      assert(findrr(RR)),
                      findrr(RR),
                      arg(1,RR,Temp),
                      Tempa = Temp,
                      assert(finda(Tempa)),
                      write_heading_1,
                      nl,nl,
                      write_fireclause,
                      retractall(findq(Q)),
                      retractall(findrr(RR)),
                      write('Look for another?  (Y/N):'),
                      get(Char),nl,
                      (Char=78 ; Char=110),    /* N or n */
                      !.


find_solution_2(Q) :-
                      find(Q),
```

```
                    call(Q),
                    assert(findq(Q)),
                    write('Solution found:    '),
                    writeln(Q),
                    nl,
                    arg(2,Q,SS),
                    Tempss = SS,
                    assert(findss(Tempss)),
                    arg(3,Q,XX),
                    Tempxx = XX,
                    assert(findxx(Tempxx)),
                    arg(4,Q,YY),
                    Tempyy = YY,
                    assert(findyy(Tempyy)),
                    arg(5,Q,ZZ),
                    Tempzz = ZZ,
                    assert(findzz(Tempzz)),
                    arg(1,Q,Temp),
                    Tempa = Temp,
                    assert(finda(Tempa)),
                    write_heading_2,
                    write_fireclause,
                    retractall(findq(Q)),
                    write('Look for another?  (Y/N):'),
                    get(Char),nl,
                    (Char=78 ; Char=110),    /* N or n */
                    !.


find_solution_3(Q) :-
                    find(Q),
                    call(Q),
                    assert(findq(Q)),
                    write('Solution found:    '),
                    writeln(Q),
                    nl,
                    arg(2,Q,XX),
                    Tempxx = XX,
                    assert(findxx(Tempxx)),
                    arg(3,Q,YY),
                    Tempyy = YY,
                    assert(findyy(Tempyy)),
                    arg(1,Q,RR),
                    Tempa = RR,
                    assert(finda(Tempa)),
                    write_heading_3,
                    write_firealarm,
                    retractall(findq(Q)),
                    write('Look for another?  (Y/N):'),
                    get(Char),nl,
```

```
                    (Char=78 ; Char=110),    /* N or n */
                    !.


find_solution_4(Q) :-
                    find(Q),
                    call(Q),
                    assert(findq(Q)),
                    write('Solution found:    '),
                    writeln(Q),
                    arg(2,Q,XX),
                    Tempxx = XX,
                    assert(findxx(Tempxx)),
                    arg(3,Q,YY),
                    Tempyy = YY,
                    assert(findyy(Tempyy)),
                    arg(4,Q,ZZ),
                    Tempzz = ZZ,
                    assert(findzz(Tempzz)),
                    arg(1,Q,RR),
                    Tempa = RR,
                    assert(finda(Tempa)),
                    write_heading_4,
                    write_safety,
                    retractall(findq(Q)),
                    write('Look for another?  (Y/N):'),
                    get(Char),nl,
                    (Char=78 ; Char=110),    /* N or n */
                    !.


find_solution_5(Q) :-
                    find(Q),
                    call(Q),
                    assert(findq(Q)),
                    write('Solution found:    '),
                    writeln(Q),
                    arg(2,Q,XX),
                    Tempxx = XX,
                    assert(findxx(Tempxx)),
                    arg(3,Q,YY),
                    Tempyy = YY,
                    assert(findyy(Tempyy)),
                    arg(4,Q,ZZ),
                    Tempzz = ZZ,
                    assert(findzz(Tempzz)),
                    arg(1,Q,RR),
                    Tempa = RR,
                    assert(finda(Tempa)),
                    write_heading_5,
                    write_exit,
```

```
                    retractall(findq(Q)),
                    write('Look for another?  (Y/N):'),
                    get(Char),nl,
                    (Char=78 ; Char=110),    /* N or n */
                    !.


find_solution_6(Q) :-
                    find(C),
                    call(Q),
                    assert(findq(Q)),
                    write('Solution found:    '),
                    writeln(Q),
                    nl,
                    arg(2,Q,XX),
                    Tempxx = XX,
                    assert(findxx(Tempxx)),
                    arg(3,Q,YY),
                    Tempyy = YY,
                    assert(findyy(Tempyy)),
                    arg(1,Q,RR),
                    Tempa = RR,
                    assert(finda(Tempa)),
                    write_heading_6,
                    write_plumbing,
                    retractall(findq(Q)),
                    write('Look for another?  (Y/N):'),
                    get(Char),nl,
                    (Char=78 ; Char=110),    /* N or n */
                    !.


find_solution_1(_) :- write('No (more) solutions'),
                      nl.

find_solution_2(_) :- write('No (more) solutions'),
                       nl.

find_solution_3(_) :- write('No (more) solutions'),
                      nl.

find_solution_4(_) :- write('No (more) solutions'),
                      nl.

find_solution_5(_) :- write('No (more) solutions'),
                      nl.

find_solution_6(_) :- write('No (more) solutions'),
                      nl.

write_heading_1 :- (heading1 ; heading2).
```

```
write_heading_2 :- heading3 , heading4, heading5 , heading6.


heading1 :-
                findx(Tempx),
                var(Tempx),
                write('Number of storeys ='),
                findl(Templ),
                write(Templ),nl,
                retractall(findl(_)).


heading2 :-
                findy(Tempy),
                var(Tempy),
                write('Building area ='),
                findl(Templ),
                write(Templ),write(' m2'),nl,
                retractall(findl(_)).



heading3 :-
                write('Number of storeys = '),
                findxx(Tempxx),
                ((nonvar(Tempxx),write(Tempxx))
                 | (var(Tempxx),write('undefined'))),nl,
                retractall(findxx(_)).

heading4 :-
                write('Building area ='),
                findyy(Tempyy),
                ((nonvar(Tempyy),write(Tempyy))
                 | (var(Tempyy),write('undefined'))),
                write(' m2'),
                 findy(Tempy),
                ((nonvar(Tempy),write(' '))
                 | (var(Tempy),write('  max'))),nl,
                retractall(findyy(_)).


heading5 :-
               findss(Tempss),
               /* finds(Temps),
               ((nonvar(Temps),write(Tempss))
                | (var(tempss),write('undefined'))),   */
               nl,
               (((Tempss = 1),write('Building is
                     unsprinklered'))
               |((Tempss = 2),write('Building is
                     sprinklered'))),
```

```
                nl,
                retractall(findss(_)).

heading6 :-
                write('Building faces '),
                findzz(Tempzz),
                ((nonvar(Tempzz),write(Tempzz))
                 | (var(Tempzz),write('undefined'))),
                write(' streets'),nl,
                retractall(findzz(_)).



write_heading_3 :-
                 write('Number of storeys = '),
                 findxx(Tempxx),
                 ((nonvar(Tempxx),write(Tempxx))
                  | (var(Tempxx),write('undefined'))),nl,
                 retractall(findxx(_)),
                 write('Occupant load  = '),
                 findyy(Tempyy),
                 ((nonvar(Tempyy),write(Tempyy))
                  | (var(Tempyy),write('undefined'))),nl,
                 retractall(findyy(_)).


write_heading_4 :-
                 write('area of room or suite = '),
                 findxx(Tempxx),
               ((nonvar(Tempxx),write(Tempxx),write('m2'))
                  | (var(Tempxx),write('undefined'))),nl,
                 retractall(findxx(_)),

                 write('distance to egress doorway  = '),
                 findyy(Tempyy),
                ((nonvar(Tempyy),write(Tempyy),write( 'm'))
                  | (var(Tempyy),write('undefined'))),nl,
                 retractall(findyy(_)),

                 write('Occupant load  = '),
                 findzz(Tempzz),
                 ((nonvar(Tempzz),write(Tempzz))
                  | (var(Tempzz),write('undefined'))),nl,
                 retractall(findzz(_)).


write_heading_5 :-
                 write('number of storeys = '),
                 findxx(Tempxx),
```

```
                ((nonvar(Tempxx),write(Tempxx))
                 | (var(Tempxx),write('undefined'))),nl,
                retractall(findxx(_)),

                write('area of room  = '),
                findyy(Tempyy),
            ((nonvar(Tempyy),write(Tempyy),write(' m2'))
                 | (var(Tempyy),write('undefined'))),nl,
                retractall(findyy(_)),

                write('travel distance  = '),
                findzz(Tempzz),
                ((nonvar(Tempzz),write(Tempzz),write(' m'))
                 | (var(Tempzz),write('undefined'))),nl,
                retractall(findzz(_)).


write_heading_6 :-
                write('Number of males  = '),
                findxx(Tempxx),
                ((nonvar(Tempxx),write(Tempxx))
                 | (var(Tempxx),write('undefined'))),nl,
                retractall(findxx(_)),
                write('Number of females  = '),
                findyy(Tempyy),
                ((nonvar(Tempyy),write(Tempyy))
                 | (var(Tempyy),write('undefined'))),nl,
                retractall(findyy(_)).


write_fireclause :-
                nl,nl,write('Fire regulation '),
                finda(Tempa),
                nl,
                fireclause(Tempa,ID1),
                writeln(ID1),
                retractall(finda(Tempa)).


write_firealarm :-
                nl,nl,write('Fire Alarm '),
                finda(Tempa),
                nl,
                firealarm(Tempa,ID1),
                writeln(ID1),
                retractall(finda(Tempa)).


write_safety :-
```